Universidade de Aveiro
2010
Departamento de Electrónica Telecomunicações e Informática

Nuno Filipe Oliveira Novo

Controlo e Gestão de Sessões Multimédia IP usando o IM-SSF

Universidade de Aveiro
2010

Departamento de Electrónica Telecomunicações e Informática

Nuno Filipe Oliveira Novo

Controlo e Gestão de Sessões Multimédia IP usando o IM-SSF

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Electrónica e Telecomunicações, realizada sob a orientação científica do Doutor Amaro Fernandes de Sousa, Professor Auxiliar do Departamento de Electrónica Telecomunicações e Informática da Universidade de Aveiro

O júri

Presidente

Doutor Aníbal Manuel de Oliveira Duarte
Professor Catedrático da Universidade de Aveiro

Vogais

Doutor Paulo Manuel Martins de Carvalho
Professor auxiliar da Escola de Engenharia da Universidade do Minho

Doutor Amaro Fernandes de Sousa
Professor auxiliar da Universidade de Aveiro

agradecimentos

Ao Professor Doutor Amaro Fernandes de Sousa, do Departamento de Engenharia de Electrónica e de Telecomunicações da Universidade de Aveiro, na qualidade de orientador, pela oportunidade concedida para a realização do presente trabalho, assim como pelo rigor, crítica permanente, sessões de trabalho conjunto e pela disponibilidade e empenho que sempre demonstrou.

Ao Mestre António Amaral pela amizade e pela colaboração que foi de extrema importância na realização desta dissertação.

À PT Inovação, pelas condições que colocaram à minha disposição.

Finalmente, gostaria de agradecer a todos os colegas da PT Inovação, pelo apoio e pela colaboração que foi de extrema importância na elaboração desta dissertação

palavras-chave Sinalização, Sessões Multimédia IP, IMS, IM-SSF, 3GPP

resumo

Na rede ***I**nternet **P**rotocol* (IP), é sempre possível um terminal comunicar com outro terminal remoto desde que seja conhecido o endereço IP destino e o número da porta da aplicação destino. Este modo de comunicação é muito limitado quer do ponto de vista do utilizador quer do ponto de vista do operador. Utilizando um protocolo de sinalização de rede IP como o ***S**ession **I**nitiation **P**rotocol* (SIP), é possível descobrir a localização do utilizador na rede e negociar os recursos a usar nos terminais numa chamada (codecs por exemplo). Com o protocolo SIP, é também dada mobilidade aos utilizadores da rede IP.

Para dar satisfação de forma integrada a requisitos como segurança, mobilidade, qualidade de serviço, gestão de sessão e tarifação foi definida, no âmbito do ***3**rd **G**eneration **P**artnership **P**roject* (3GPP), a arquitectura ***I**P **M**ultimedia **S**ubsystem* (IMS) baseada no protocolo SIP, permitindo suporte de serviços e aplicações multimédia em redes IP.

Sob o ponto de vista dos operadores é importante aplicar todo o trabalho desenvolvido no controlo e gestão de chamadas ***G**lobal **S**ystem for **M**obile Communications* (GSM), a sessões IP. É neste contexto que se enquadra o ***I**P **M**ultimédia **S**ervice **S**witching **F**unction* (IM-SSF). Na arquitectura IMS, enquadra-se na categoria dos ***A**pplication **S**ervers* (AS's) e possibilita que o controlo e gestão de sessões multimédia IP sejam feitos através da lógica GSM existente. Existem vários tipos de AS, os que funcionam como ponto terminal da chamada, os que funcionam apenas como um ponto de transporte na rede e ainda os ***B**ack to **B**ack **U**ser **A**gent* (B2BUA) 3GPP TS 23.218 onde podem interligar duas chamadas diferentes.

O IM-SSF opera como um B2BUA e proporciona o controlo de sessões multimédia IP participando no estabelecimento de sessão na rede IMS.

O sistema IM-SSF, definido em 3GPP TS 23.278, pode ser visto como um servidor de aplicações SIP que implementa a porta de ligação entre o protocolo SIP da interface ***I**MS **S**ervice **C**ontrol* (ISC) e o protocolo ***C**ustomized Application for Mobile Network Enhanced Logic **A**pplication **P**art* (CAP) que é utilizado na interface entre o IM-SSF e as funções de controlo de serviço

Esta dissertação surge no seguimento do trabalho efectuado na PT Inovação que proporcionou os meios necessários para o desenvolvimento e teste de um módulo IM-SSF com vista à sua integração nas soluções da empresa.

keywords Signaling, IP Multimedia Sessions, IMS, IM-SSF, 3GPP

abstract

In the network ***I**nternet **P**rotocol* (IP), it is always possible for a terminal to communicate with other remote terminal since the destination IP address and port number is known. This kind of communication it's very limited for the users and even for the operator. Using a signalization protocol like the ***S**ession **I**nitiation **P**rotocol* (SIP) it's possible to find out the location of the user in the network and negotiate the resources that will be used by the terminals in the call (codecs for example). With SIP protocol it's also given mobility to users of the IP network.

To satisfy requirements like security, mobility, quality of service, session management and charging it was defined by ***3**rd **G**eneration **P**artnership **P**roject* (3GPP), the ***I**P **M**ultimedia **S**ubsystem* (IMS) architecture, based in the SIP protocol, allowing service support and multimedia applications in IP networks.

On the point of view of operators its important apply all the work developed in session and management of sessions in ***G**lobal **S**ystem for **M**obile Communications* (GSM), to IP sessions. It's in this context that ***I**P **M**ultimedia **S**ervice **S**witching **F**unction* (IM-SSF) fits. In IMS architecture IM-SSF is an ***A**pplication **S**ervers* (AS's) and allows that the control and management of IP multimedia sessions could be made using the existing GSM control logic. There are three types of AS: the ones that function like a terminal point of the call, the ones that function like a transport point in the network and the ***B**ack to **B**ack **U**ser **A**gent* (B2BUA) 3GPP TS 23.218 that can interconnect two different calls.

The IM-SSF operates like a B2BUA and furnishes the session control in IP multimedia, by participating in the IMS session establishment.

The IM-SSF system, defined in 3GPP TS 23.278 can be seen like a server of SIP applications that implements the gateway between SIP protocol of ***I**MS **S**ervice **C**ontrol* (ISC) interface and the ***C**ustomized Application for Mobile Network Enhanced Logic **A**pplication **P**art* (CAP) protocol that's used in the interface between IM-SSF and the functions that control the service.

Índice

Índice de figuras

Índice de Tabelas

Acrónimos

3GPP	3rd Generation Partnership Project
AAA	Authentication, Authorization, and Accounting
AS	Application Server
AVP	Attribute Value Pair
BGCF	Breakout Gateway Control Function
CAMEL	Customized Applications for Mobile Network Enhanced Logic
CAP	CAMEL Application Part
CDR	Call Detail Records
CSCF	Call Session Control Function
CSGW	CAMEL Signaling GateWay
DHCP	Dynamic Host Configuration Protocol
DSCP	Data Service Control Point
GSM	Global System for Mobile Communications
gsmSCF	GSM Service Control Function
HLR	Home Location Register
HSS	Home Subscriber Server
I-CSCF	Interrogating Call Session Control Function
IM-CSI	IP Multimedia CAMEL Subscription Information
IMS	IP Multimedia Subsystem
IM-SSF	IP Multimedia Service Switching Function
IP	Internet Protocol
ISC	IMS Service Control
ISDN	Integrated Services Digital Network
ISUP	ISDN User Part
MAP	Mobile Application Part
MGCF	Media Gateway Control Function
MRFC	Multimedia Resource Function Controller

MTP	Message Transfer Part
NAS	Network Access Server
P-CSCF	Proxy Call Session Control Function
PDP	Packet Data Protocol
PSTN	Public Switched Telephone Network
RDIS	Rede Digital com Integração de Serviços
REL	Red Hat Enterprise Linux
RTDAP	Real Time Data Application Part
RTTAP	Real Time Transaction Application Protocol
S-CSCF	Serving Call Session Control Function
SCCP	Signalling Connection Control Part
SDL	Specification and Description Language
SDP	Session Description Protocol
SIP	Session Initiation Protocol
SLF	Subscription Locator Function
SS7	Signalling System Number 7
TCAP	Transaction Capabilities Application Part
THIG	Topology Hiding Inter-network Gateway
TUP	Telephone User Part
UA	User Agent
UMTS	Universal Mobile Telecommunications System
URI	Uniform Resource Indentifier
VLR	Visitor Location Register
VoIP	Voice over IP

Capítulo 1 - Introdução

1.1 *Motivação e enquadramento*

Na rede *Internet Protocol* (IP) é sempre possível um terminal comunicar com outro terminal remoto desde que seja conhecido o endereço IP destino e o número da porta da aplicação destino. No entanto, este modo de comunicação é muito limitado quer do ponto de vista do utilizador quer do ponto de vista do operador que pretenda oferecer mobilidade aos utilizadores, controlo de sessões de voz, vídeo, e outros serviços para o utilizador. Utilizando um protocolo de sinalização, como o *Session Initiation Protocol* (SIP), é possível descobrir a localização do utilizador na rede e ao mesmo tempo negociar os recursos a usar nos terminais numa chamada (*codec's* por exemplo). Com o protocolo SIP, é também possível a mobilidade dos utilizadores na rede IP, ou seja, é possível sempre identificar um determinado utilizador com base no seu identificador SIP permitindo que independentemente do endereço IP, o utilizador possa ser sempre identificado correctamente.

Para dar satisfação de forma integrada a requisitos como segurança, mobilidade, qualidade de serviço, gestão de sessão e tarifação foi definida, no âmbito do *3rd Generation Partnership Project* (3GPP), a arquitectura *IP Multimedia Subsystem* (IMS) baseada no protocolo SIP, permitindo suporte de serviços e aplicações multimédia em redes IP.

Sob o ponto de vista dos operadores é importante poder aplicar às sessões multimédia IP todo o trabalho desenvolvido no controlo e gestão de chamadas *Global System for Mobile Communications* (GSM). É neste contexto que se enquadra o *IP Multimedia Service Switching Function* (IM-SSF). Na arquitectura IMS, o IM-SSF enquadra-se na categoria dos *Application Servers* (AS's) e possibilita que o controlo e gestão de sessões multimédia IP sejam feitos através da lógica GSM existente.

Esta dissertação surge no seguimento do trabalho efectuado na PT Inovação que proporcionou os meios necessários para o desenvolvimento do módulo IM-SSF e consequentemente a possibilidade da realização desta dissertação.

1.2 *Conceitos Fundamentais*

Com o aumento crescente das plataformas, serviços e equipamentos que usam conectividade IP, torna-se cada vez mais uma necessidade conseguir utilizar os serviços que até agora eram disponibilizados nas redes tradicionais de telecomunicações. Com isto surge a necessidade de estabelecer ligações IP com garantias de qualidade de serviço (QoS) para serviços de tempo real. Com a transposição das telecomunicações para a rede IP, ganham-se algumas vantagens tais como a interacção directa com a Internet (explorando serviços aí disponíveis) e a utilização de protocolos IP já existentes para o estabelecimento das chamadas e comunicação entre os intervenientes. Para colmatar esta necessidade foi definida a arquitectura IMS. Esta arquitectura foi projectada para o funcionamento em redes IP, tendo a capacidade de fornecer todo o tipo de serviços existentes ou novos sem alterar a infra-estrutura. É na categoria de fornecedor de serviços que o IM-SSF se encaixa. O IM-SSF proporciona o controlo de sessões multimédia IP participando no estabelecimento de sessão na rede IMS. O IM-SSF funciona como *gateway* inteligente que, através do contacto com elementos da rede GSM, consegue fornecer os mesmos serviços já proporcionados pela rede GSM.

O controlo na rede IMS é efectuado utilizando o protocolo SIP [RFC 3261]. Este protocolo foi escolhido como parte integrante da rede IMS pelo facto de ser um protocolo IP simples, projectado para o estabelecimento de sessões onde todos os tipos

de dados (voz, vídeo, texto) usam o mesmo formato de transmissão. No contacto com o *Home Location Register* (HLR), ou *Home Subscriber Server* (HSS) como é designado na rede IMS, a comunicação é efectuada utilizando o protocolo *Mobile Application Part* (MAP) ou Diameter.

Uma sessão SIP é estabelecida entre dois clientes através da troca de mensagens específicas. O chamador envia a primeira mensagem, o "INVITE", para um elemento SIP conhecido (*proxy gateway*). Este elemento encaminha a mensagem até ao destino, podendo a mensagem eventualmente passar por outros elementos SIP até atingir o destino. Quando a mensagem "INVITE" chega ao destino, fica estabelecido o percurso SIP. O destinatário responde pelo percurso SIP estabelecido com um "200 OK" que confirma a recepção e aceitação da ligação. Ao receber esta mensagem, o chamador responde com um "ACK" que reconfirma que a ligação de sinalização está em boas condições e a ligação de dados pode ser estabelecida. Após este ponto, ambos os intervenientes já sabem para onde dirigir os seus pacotes de dados media, estando assim, do ponto de vista da sinalização SIP, a sessão estabelecida. Para terminar uma sessão estabelecida, um dos intervenientes envia a mensagem de "BYE" que agora segue por um percurso marcado por alguns intervenientes aquando do encaminhamento do "INVITE". Esta mensagem ao chegar ao destino é respondida com um "200 OK" que finaliza a sessão estabelecida.

O IM-SSF é um elemento interveniente no encaminhamento do "INVITE" e por consequência da sessão. Com isto consegue obter o controlo da sessão através da manipulação das mensagens que por si passam estabelecendo, por vezes, novas ligações com outros elementos. Através do contacto com o HSS, o IM-SSF consegue obter informação necessária dos clientes intervenientes na sessão e por conseguinte fornecer os serviços específicos a cada um deles. Estes serviços são fornecidos através da consulta do módulo *GSM Service Control Function* (*gsmSCF)* que indicará o que se deve fazer para que o cliente possa desfrutar dos serviços a que tem direito. O IM-SSF tem acesso apenas à sinalização da mensagem; por ele não passam os pacotes de media embora este

os possa controlar indirectamente através do uso de outro elemento da rede IMS (o *Multimedia Resource Function Controller*).

1.3 *Objectivos*

Esta dissertação tem como objectivos:

i) estudar as normas vigentes que suportam o IM-SSF

ii) estudar os protocolos que suportam as interfaces entre o IM-SSF e as diversas entidades associadas

iii) implementar o módulo IM-SSF

iv) testar o desempenho das funcionalidades do módulo.

1.4 *Estrutura da dissertação*

No capítulo 2 é feita uma pequena introdução à arquitectura IMS que expõe alguma linguagem e pormenores mais técnicos relevantes para a melhor compreensão do módulo IM-SSF e a sua função na arquitectura IMS.

O capítulo 3 descreve o desenvolvimento do módulo IM-SSF. Neste capítulo são descritos alguns processos utilizados no módulo e comparam-se algumas soluções possíveis para resolver determinadas situações específicas da implementação.

No capítulo 4 são apresentados os cenários de teste desenvolvidos para demonstrar o correcto funcionamento do módulo e o seu desempenho. Nos cenários de desempenho são anotadas algumas variáveis com as quais se pode realizar uma imagem da eficiência da solução desenvolvida.

Por último são retiradas algumas conclusões de todo o trabalho realizado tendo em conta o que está especificado, foi feito e se planeia fazer ao nível de novas funcionalidades.

Capítulo 2 - Análise normativa

Neste capítulo é apresentada a arquitectura IMS, os seus principais componentes e os protocolos principais usados. Pretende-se clarificar os pontos essenciais que serão necessários para a compreensão da acção do IM-SSF. Deste modo é necessário compreender o IMS pois é nesta rede que o IM-SSF opera.

2.1 *IMS*

Para melhorar a compreensão dos vários módulos que constituem o IMS, é necessário conhecer como são estabelecidas chamadas no IMS. Para este propósito é necessário entender primeiro quais os protocolos usados, quais os módulos que os usam e como estes empregam os protocolos para interagir entre si e assim formar a arquitectura IMS.

2.1.1 Protocolos

Os protocolos usados pelo IMS são o protocolo SIP, que é o responsável pelo estabelecimento de sessões no IMS e o protocolo Diameter, que é o responsável pela comunicação com as bases de dados que armazenam os dados dos clientes.

2.1.1.1 SIP

O protocolo ***S**ession **I**nitiation **P**rotocol* (SIP), definido na RFC 3261, foi desenvolvido com o objectivo de suportar o estabelecimento de sessões multimédia entre utilizadores da rede IP. Na linguagem SIP, as mensagens de sinalização que iniciam acções tomam o

nome de *methods* e a cada *method* podem corresponder várias respostas. Uma sessão SIP corresponde à chamada tradicional entre utilizadores da rede IP.

As trocas de sinalização para o estabelecimento, terminação ou modificação de uma sessão são denominadas transacções SIP. Uma transacção SIP é constituída por um pedido (*request*) seguido de uma ou mais respostas informativas, e por uma ou mais respostas finais. Por exemplo, no estabelecimento de uma sessão *Voice over IP* (VoIP) entre utilizadores, é feito o pedido de estabelecimento da sessão com a utilização do *method* "INVITE". Quando a campainha do chamado toca, o equipamento chamado responde com a resposta informativa "180 RINGING" e quando o utilizador atende é enviada a mensagem de resposta final "200 OK" indicando sucesso no estabelecimento da chamada. O utilizador que enviou o *"INVITE"* assinala a recepção da mensagem "200 OK" com a mensagem "ACK" que encerra a transacção de estabelecimento da chamada.

Para o transporte das mensagens SIP, pode ser usado o protocolo TCP ou UDP. Se for usado o protocolo UDP, o protocolo SIP para garantir a chegada das mensagens ao destino, repete o envio das mensagens temporizadamente até que seja recebida a confirmação do destino. Para todos os pedidos (excepto o "INVITE") há repetição das mensagens até chegar a resposta definitiva. No caso do pedido "INVITE", a resposta final é dependente da reacção do utilizador chamado a atender a chamada. A resposta final pode demorar muito tempo e isso corresponderia a uma sobrecarga da rede com mensagens de "INVITE*"* desnecessárias. Sendo assim, a mensagem de "INVITE" pára de ser enviada logo que seja recebida uma resposta informativa (ex. "180 RINGING"*).*

O toque da campainha do chamado é assinalado ao chamador através da tonalidade de chamar. Se a mensagem OK se perder o chamador continuaria a receber a tonalidade de chamar apesar de o chamado já estar em "linha". Para evitar esta situação, que originaria a falha do estabelecimento da sessão, é utilizada a mensagem "ACK". A mensagem de "200 OK" é repetida até que o chamado receba a mensagem de "ACK".

As respostas informativas também podem ser confirmadas pelo receptor. É utilizada para isso a mensagem de "PRACK". Esta mensagem é enviada quando a mensagem de resposta informativa transporta explicitamente o pedido de confirmação.

A Figura 2-1 representa a sequência das mensagens utilizadas no estabelecimento da sessão.

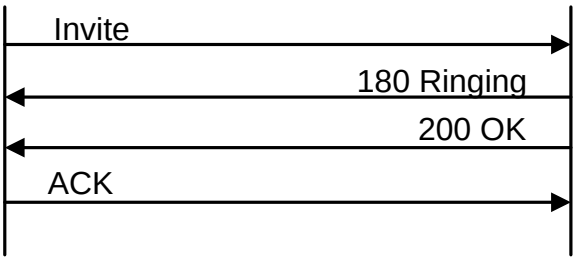

Figura 2-1: Estabelecimento directo de uma chamada SIP

O modo de funcionamento do SIP tal como descrito até aqui pressupõe que o chamador conhece o endereço IP e a porta onde o chamado recebe as mensagens de sinalização. Só assim é possível enviar a mensagem de "INVITE" directamente para o chamador.

O envio directo da mensagem de "INVITE" para o chamador apresenta várias limitações. Ter que ser conhecido o endereço IP e a porta do utilizador impede a mobilidade dos utilizadores como serviço prestado pelo operador e desaproveita todos os mecanismos de encaminhamento e descoberta de endereços já implementados na rede IP. Além disso, a troca de sinalização entre chamador e chamado não permite a possibilidade de controlar o acesso à rede dos utilizadores para efeitos de gestão de recursos, identificação, autenticação e tarifação.

Para garantir a flexibilidade necessária no estabelecimento das sessões e a mobilidade dos utilizadores, a rede SIP integra como seus componentes os servidores *Proxy*, *Redirect*, *Registration* e *Location.*

O SIP corre sobre o protocolo IP e baseia as suas mensagens no protocolo HTML. Este tipo de mensagens privilegiam a leitura do utilizador humano devido ao seu pequeno texto descritivo do erro que para nós humanos é-nos mais fácil compreender pois do ponto de vista da máquina é apenas uma questão de hardware. Isto torna a análise do

SIP mais perceptível ao nível humano visto que, quase todos os que já tiveram algum contacto com a Internet conhecem algumas das suas mensagens de erro.

O SIP define 3 tipos base, o proxy, o *User Agent Server* (UAS) e o *User Agent Client* (UAC). Dentro dos user agents existe um tipo que é uma junção do UAS mais o UAC que se chama o *Back to Back User Agent* (B2BUA).

Na arquitectura IMS como proxy SIP temos o I-CSCF, S-CSCF e o P-CSCF. Os AS podem ser de todos os tipos SIP base mais o tipo B2BUA, sendo este o tipo de AS que define o IM-SSF.

2.1.1.2 Diameter

O protocolo Diameter, definido na [RFC 3588], é um protocolo de *Authentication, Authorization and Accounting* (AAA) desenvolvido para providenciar de uma forma genérica os requisitos de um protocolo AAA requeridos pelo [AAA **Jaques**].

O protocolo Diameter define as suas mensagens como conjuntos de *Attribute Value Pairs* (AVP). Os AVPs são formados por um cabeçalho mais um campo de dados. O cabeçalho é composto da seguinte forma:

- Código – Identificador numérico do AVP.
- Flags – Informação para o receptor do AVP, indicando como este AVP deve ser administrado.
- Tamanho – Indica tamanho do AVP em bytes, incluindo o cabeçalho e os dados.
- Vendor Id – Identificador numérico que identifica a identidade de quem definiu o AVP (por exemplo, o *vendor Id* do 3GPP é o 10415).

Um AVP pode ser um tipo simples (ex: string) ou então ele próprio pode ser composto por um conjunto de AVPs. Cada AVP é definido inequivocamente através do seu código e do seu *vendor Id*. Cada mensagem Diameter possui AVPs que são fixos, outros obrigatórios e alguns opcionais (ver Figura 2-2). Com isto é possível garantir que duas entidades conseguem comunicar entre si desde que respeitem os AVP fixos e obrigatórios. A norma permite que a mensagem contenha outros AVPs no campo de AVPs extra. Se ambas as entidades comunicantes estiverem de acordo, podem ainda

transportar informação extra num outro AVP que não estando definido para a mensagem pode ser utilizado nessa mensagem no campo de AVPs extra.

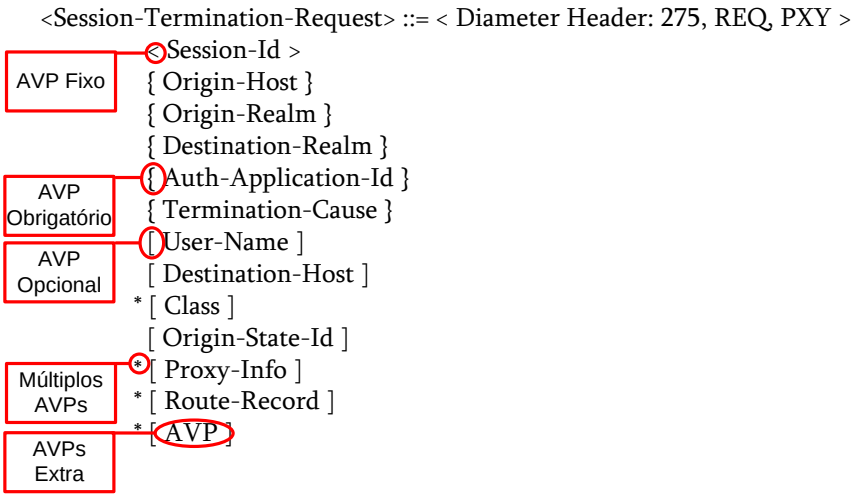

Figura 2-2: Mensagem Diameter

A Figura 2-2 apresenta um exemplo típico da definição de uma mensagem Diameter, onde são definidos pelo nome os AVPs utilizados na mensagem (por exemplo o Session-Id) Como foi dito anteriormente um AVP é identificado pelo seu código e vendor-Id assim este nome do AVP é apenas válido dentro da especificação da mensagem ou então tem de ser indicado o local onde está definido o AVP que se está a utilizar. Como é ilustrado na figura, a simbologia para os avps é a seguinte:

< > indica um AVP fixo, ou seja o AVP tem sempre de vir nesta posição dentro da mensagem neste exemplo é sempre o primeiro AVP a aparecer.

{ } indica um AVP obrigatório, ou seja para a mensagem ser considerada correcta este AVP tem de estar presente.

[ ] indica um AVP opcional.

* indica que podem existir múltiplos AVPs deste tipo.

*[AVP] indica que qualquer AVP definido pode ser acrescentado na mensagem como AVP opcional.

Qualquer nó pode iniciar um pedido. Neste ponto de vista, o Diameter é um protocolo *peer-to-peer.* Duas entidades comunicantes são idênticas entre si (ao nível do

estabelecimento de ligação) e nas mensagens de ligação negoceiam a versão e outros dados que permitem que ambas se conheçam.

O protocolo Diameter providencia as seguintes funções:

- Entrega d'AVPs
- Negociação das potencialidades (na fase inicial é negociada a versão e outros dados que permitem aos pares falar entre si)
- Notificação d'erros
- Expansibilidade através da adição de novos comandos e AVPs (requeridos em [AAA **Jaques**])
- Serviços básicos necessários para aplicações, como o tratamento de sessões do utilizador e taxação.

Todos os dados entregues pelo protocolo estão na forma de um AVP. Alguns valores destes AVPs são usados pelo protocolo Diameter, enquanto outros entregam dados associados com aplicações específicas que utilizam o Diameter. Os AVPs podem ser adicionados por qualquer ordem nas mensagens Diameter (excepto os fixos conforme já foi explicado), desde que os AVPs obrigatórios e fixos estejam presentes a mensagem está correcta. Os AVPs são usados pelo protocolo Diameter para suportar as seguintes características requeridas:

- Transporte de informação de autenticação de um utilizador, com o propósito de permitir ao servidor Diameter autenticar o utilizador em questão.
- Transporte de informação específica de serviços entre clientes e servidores, permitindo aos pares decidir se o acesso do utilizador será permitido.
- Troca de informação sobre a utilização de recursos que podem ser usados para propósitos de taxação, planeamento de capacidade, etc.
- Passagem de mensagens Diameter através de uma hierarquia de servidores.

O protocolo base Diameter pode ser usado só por si para propósitos de taxação, ou pode ser usado com uma aplicação Diameter como o Mobile IPv4 [Diameter Perkins], ou acesso à rede [NASREQ Haag].

O Diameter define quatro entidades base: o cliente, o agente, o servidor e o nó.

Um cliente Diameter é um dispositivo que no limiar da rede efectua o controlo de acesso como um *Network Access Server* (NAS) ou um *Foreign Agent* (FA). Um cliente Diameter gera mensagens Diameter para pedir serviços de autenticação, autorização e taxação para o utilizador. Um agente Diameter é um nó intermédio que efectua apenas a retransmissão das mensagens (*proxis*, agentes de redireccionamento e de *relay*.) Um servidor Diameter efectua a autenticação e/ou autorização do utilizador. Um nó Diameter pode funcionar como agente para alguns pedidos e funcionar como servidor para outros.

O protocolo Diameter também suporta pedidos iniciados no servidor, tal como pedidos para abortar o serviço de um determinado utilizador.

No IMS, o Diameter é o protocolo escolhido para transportar informação de/para o HSS, pois, devido à sua versatilidade nas mensagens, cada mensagem pode transportar muita informação, algo que favorece o transporte de quantidades maciças de informação. Na arquitectura IMS como cliente Diameter temos o I-CSCF, S-CSCF e os AS, como servidor Diameter temos o HSS e como agente temos o SLF (estes elementos serão introduzidos na secção seguinte).

2.1.2 Arquitectura IMS

Na arquitectura IMS, existem vários módulos responsáveis por sinalizar as sessões estabelecidas entre dois ou mais clientes. Entre os módulos mais importantes distinguem-se o *Proxy Call Session Control Function* (P-CSCF), *Interrogating* CSCF (I-CSCF), *Serving* CSCF (S-CSCF) e o HSS que fazem parte do núcleo da rede IMS. Igualmente importantes são os AS que, não pertencendo ao núcleo IMS, merecem especial atenção por fornecerem serviços importantes para o funcionamento da rede de uma forma eficaz e produtiva do ponto de vista de negócio.

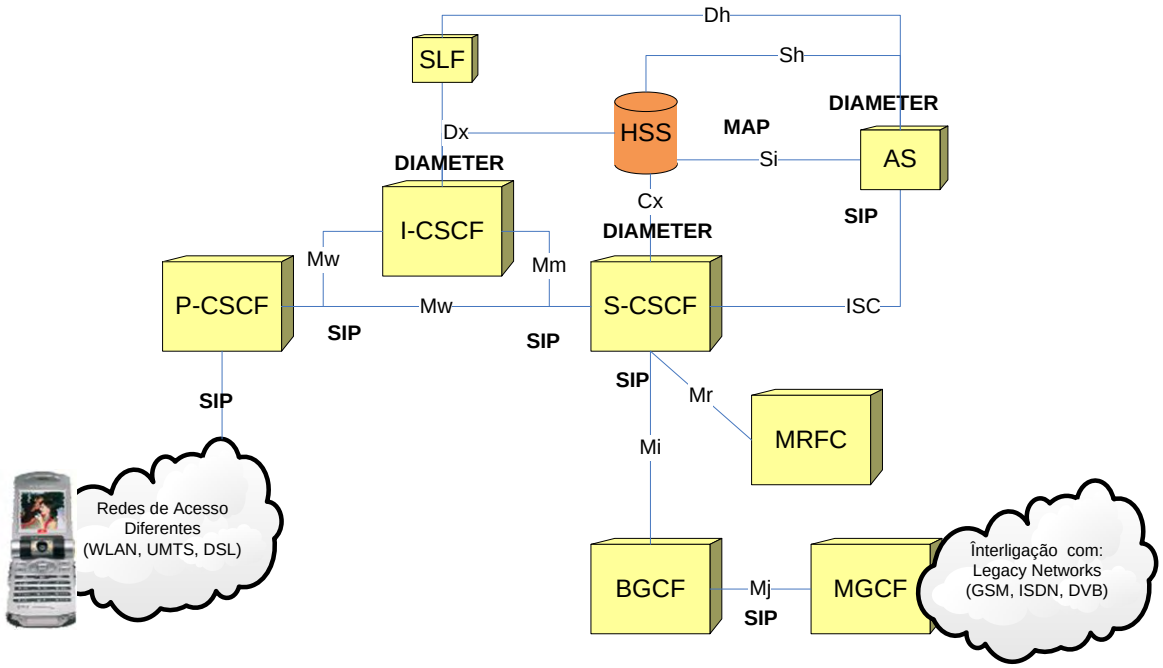

Figura 2-3: Arquitectura IMS

Na Figura 2-3 é apresentado um esquema da arquitectura IMS onde são apresentados os seus elementos constituintes e as interfaces definidas entre eles. Para se compreender como os elementos da rede IMS funcionam e qual o seu papel na rede é necessário considerar um exemplo simples. O exemplo escolhido para mostrar como a rede funciona é o processo de registo que é efectuado sempre que um terminal se liga. Quando um terminal se liga, a primeira coisa que pede à rede local é o endereço de um P-CSCF que possa usar. Para isto é usado o protocolo *Dynamic Host Configuration Protocol (*DHCP). Após obtenção do endereço do P-CSCF, o cliente já pode iniciar o processo de registo na rede IMS. O cliente envia para o P-CSCF a mensagem SIP "REGISTER". O P-CSCF ao receber o "REGISTER" não sabe para onde o enviar mas sabe qual o cliente origem do "REGISTER" que é o cliente que se pretende registar. Desta forma, através da análise do endereço SIP origem o P-CSCF sabe qual é a operadora IMS à qual o cliente pertence. Neste ponto, o P-CSCF reenvia a mensagem "REGISTER" para o I-CSCF da operadora IMS do cliente. O I-CSCF é o ponto de entrada externo e, como pertence à rede do cliente da sessão, necessita agora de saber

qual o S-CSCF que irá servir este cliente. Para isso, é necessário aceder aos dados do cliente em causa para que lhe seja atribuído um S-CSCF. Estes dados estão guardados no HSS mas devido à enormidade de registos que podem existir num HSS, é possível existir mais do que um HSS numa mesma rede IMS. Assim é necessário que alguém consiga indicar o HSS onde se encontram os dados de determinado assinante. É precisamente esta a função do *Subscription Locator Function* (SLF). Então o I-CSCF usando o Diameter, contacta o SLF para saber qual o HSS onde se encontram os dados do assinante. O SLF indica ao I-CSCF qual o HSS que deve ser consultado. Com esta informação, o I-CSCF contacta o HSS pedindo-lhe um S-CSCF para servir o assinante. O HSS pode enviar-lhe o endereço de um S-CSCF ou uma lista deles. Ao receber esta informação, o I-CSCF tenta contactar um S-CSCF para lhe reenviar o "REGISTER" do cliente. Neste ponto, o S-CSCF contacta o HSS para indicar que o assinante vai utilizar este S-CSCF para as comunicações. Além desta indicação, o S-CSCF também pede os *Initial Filter Criteria* (IFCs) do assinante que está a registar. Estes IFC são uma lista de critérios que, quando cumpridos, indicam o endereço de um AS que deve ser contactado. O S-CSCF analisa os IFCs pois o assinante pode ter algum serviço que é utilizado no registo (ex: o envio de uma SMS dando-lhe as boas vindas). Caso não exista nenhum serviço para contactar na fase de registo (ou após esse serviço ser realizado) o S-CSCF envia uma resposta para o cliente. No caso de o registo ter sido efectuado com sucesso é enviada a mensagem "200 OK". Esta mensagem percorre o caminho inverso efectuado pela mensagem REGISTER, ou seja, passa primeiro pelo I-CSCF e depois pelo P-CSCF antes de chegar ao terminal do cliente. No caso de o registo não ter sido aceite pelo sistema, o S-CSCF envia uma mensagem de erro (mensagens da família 300, 400, 500) que percorre o mesmo percurso que é percorrido pelo "200 OK" no caso de sucesso.

2.1.2.1 P-CSCF

O *Proxy Call Session Control Function* (P-CSCF) é o ponto de entrada na rede IMS. É através do P-CSCF que o cliente consegue aceder à sua rede e efectuar chamadas. Para

além de outras funções que não cabem no âmbito desta descrição, o P-CSCF providencia as seguintes funções ([3GPP TS 23.228 V6.8.0],[3GPP TS 24.229 V6.14.0]):

- É o primeiro ponto de contacto do equipamento do utilizador com o IMS. Encaminha o pedido de registo para o I-CSCF, após o qual encaminha as mensagens SIP entre o equipamento do utilizador e o I-CSCF /S-CSCF.
- Funciona como um proxy SIP definido em [RFC 3261], ou seja aceita pedidos e ou serve-os internamente ou encaminha-os com a possibilidade de alguma manipulação de alguns parâmetros.
- Pode comportar-se como *User Agent (UA)*, definido em [RFC 3261], ou seja, pode em casos anormais terminar uma chamada e independentemente gerar transacções SIP.
- É descoberto via DHCP durante o processo de registo ou o endereço pode ser enviado com a activação do contexto *Packet Data Protocol* (PDP).
- Pode modificar o *Uniform Resource Indicators* (URI) de pedidos que vai encaminhar de acordo com as regras locais do operador (ex: efectua análise de dígitos, detecta números locais de serviço).
- Detecta e encaminha chamadas de emergência para o S-CSCF local.
- Gera informação de taxação offline (*Charging*), ou seja é capaz de guardar informação sobre uma sessão, chamador e chamado, duração da sessão entre outros. Esta informação só é gerada e processada após o término da sessão.
- Mantém uma associação segura (*security association*) com o equipamento do utilizador, fornecendo também segurança na direcção do S-CSCF.

O P-CSCF possui uma interface *Mw* com a qual consegue comunicar com outros elementos da rede IMS utilizando o protocolo SIP.

2.1.2.2 I-CSCF

O *Interrogating Call Session Control Function* (I-CSCF) providencia as seguintes funções:

- É o ponto de contacto dentro da rede de um operador para todas as ligações destinadas a um cliente desta rede, ou para um cliente desta rede que se encontre em *roaming* (P-CSCF não pertencente à mesma rede que o I-CSCF).
- Pode ser visto como uma espécie de *firewall* entre uma rede IMS externa e a rede IMS interna de um operador. Podem existir múltiplos I-CSCF dentro duma rede de um operador.
- Atribui um S-CSCF a um utilizador que está a efectuar o registo.
- Encaminha um pedido SIP de uma outra rede para o S-CSCF.
- Obtém do HSS o endereço do S-CSCF.
- Tal como o P-CSCF o I-CSCF é capaz de armazenar informação de Taxação, para que seja possível efectuar a taxação offline com base nessa informação.
- Ao funcionar como ponto de entrada de um subsistema IMS externo, o operador pode optar por esconder a configuração, *Topology Hiding Internetwork Gateway* (THIG), capacidade e topologia da sua rede IMS interna.

O I-CSCF possui uma interface na qual fala SIP e outra na qual fala Diameter. A interface SIP permite que o I-CSCF contacte o P-CSCF e o S-CSCF. A interface Diameter é utilizada sobretudo para contactar o HSS e, com este meio de comunicação, actualizar e receber informação sobre o cliente (estado do utilizador, S-CSCF que serve o cliente, etc.)

2.1.2.3 S-CSCF

O *Serving Call Session Control Function* (S-CSCF) tem as seguintes funcionalidades:

- Efectua o controlo dos serviços da sessão para o terminal. Dentro da rede do operador, diferentes S-CSCF podem ter diferentes funcionalidades.
- Na fase de registo, é associado um S-CSCF a cada utilizador. Este S-CSCF mantém o estado das sessões deste utilizador e tem o controlo dessas sessões.

- Actua como um Registrar definido em [RFC 3261], ou seja, aceita pedidos de registo e torna essa informação disponível através do servidor de localização (ex. HSS).
- Pode comportar-se como um proxy SIP ou um equipamento terminal (*User Agent*) como definido por [RFC 3261].
- Interage com plataformas de serviços (***A**plication **S**ervers*) para o suporte de serviços.
- Obtém o endereço do I-CSCF destino, com base no número digitado ou no SIP URI. Isto significa que através da análise do SIP URI destino da sessão o S-CSCF consegue saber qual o operador que serve esse destino. Com uma consulta a um servidor DNS facilmente obtém o ponto de entrada nessa rede IMS (I-CSCF).
- Encaminha em nome do equipamento do utilizador (***U**ser **E**quipment*) o pedido ou resposta para o P-CSCF ou o I-CSCF caso este seja utilizado no caso de *roaming*. Isto significa que no caso em que se tem uma sessão estabelecida, se o S-CSCF do chamador perder a ligação com o S-CSCF do destino, envia uma mensagem de BYE para o chamador como se tivesse sido o destino a enviá-la.
- Gera informação de taxação offline, tal como o I-CSCF e o P-CSCF. No entanto o S-CSCF como pode contactar um AS pode através deste efectuar a taxação online ou seja é capaz de cobrar os custos da sessão enquanto esta está a decorrer (por exemplo serviço de pré-pago que é descontado um montante ao saldo, previamente carregado pelo cliente, até este acabar ou a sessão terminar).

O S-CSCF, como outros elementos da rede IMS, também possui uma interface SIP que utiliza para comunicar com os outros elementos informação de sinalização relativa a sessões SIP. Além da interface SIP, também possui uma interface Diameter que é utilizada para a comunicação com o HSS. A interface Diameter possibilita ao S-CSCF o registo do cliente no HSS e a obtenção do IFC. O IFC é um género de filtro SIP onde se definem as mensagens que despoletam determinados ASs.

2.1.2.4 HSS

O *Home Subscriber Server* (HSS) é uma combinação do *Universal Mobile Telecommunications System* (UMTS)/GSM *Home Location Register* (HLR), *Visitor Location Register* (VLR) e as funcionalidades necessárias de registo para o IMS. O HSS fornece as seguintes funcionalidades:

- Identificação do utilizador, tratamento de dígitos e informação de endereço.
- Informação de segurança do utilizador: informação de controlo de acesso à rede para autenticação e autorização.
- Informação de localização do utilizador ao nível dos sistemas que o utilizador usa para aceder à rede IMS (P-CSCF e S-CSCF). O HSS trata do registo e guarda a informação de localização entre sistemas, ou seja o endereço do P-CSCF e S-CSCF que servem o utilizador.
- O perfile do utilizador (serviços, informação especifica de serviços, …) [3GPP TS 23.228 V6.8.0]

2.1.2.5 BGCF e MGCF

Considere-se que o S-CSCF, possivelmente em conjunto com um *Aplication Server*, determina que a sessão deve ser encaminhada para a *Public Switched Telephone Network* (PSTN). Neste caso, o S-CSCF encaminha a informação para o *Breakout Gateway Control Function* (BGCF) existente na sua rede. O BGCF selecciona a rede na qual a interligação deve ocorrer baseando-se nas políticas locais. Se o BGCF decide que a interligação deve ocorrer na rede que ele opera selecciona o *Media Gateway Control Function* (MGCF) que irá efectuar a interligação da rede PSTN com a rede IMS. Caso o BGCF decida que a interligação deve-se efectuar noutra rede, encaminha o pedido para o BGCF na rede seleccionada.

O MGCF fornece a conversão protocolar entre o SIP e o *ISDN User Part* (ISUP) (interligação com a PSTN). Este elemento controla o *media gateway* responsável por efectuar a conversão de dados.

2.1.2.6 MRFC

O *Multimedia Resource Function Controller* (MRFC) providencia as funções de *multiparty* e conferência multimédia [3GPP TS 23.228 V6.8.0]. O MRFC consegue controlar o *media* da conversa inserindo-se como nó intermédio quando a sessão está a ser estabelecida. Desta forma possibilita a conversação entre grupos, mediando quem tem o controlo para falar. Ao conseguir aceder directamente ao media, o MRFC, consegue incluir-se na ligação de dados como um elemento intermédio ou final dessa ligação e assim consegue enviar mensagens de voz, vídeo, etc. Com a ajuda de um AS que controla a sessão ao nível da sinalização SIP, é possível fornecer os mais variados serviços ao utilizador, tais como *voice mail*, conferência e até enviar um sinal sonoro quando é atingido o fim do saldo. Todos estes cenários são exemplos comuns da utilização do MRFC.

2.1.2.7 AS

O *Aplication Server* (AS) é o módulo responsável por fornecer serviços avançados na rede IMS. A rede IMS por si só consegue fornecer os serviços básicos de estabelecimento e manutenção de uma sessão. Os AS fornecem serviços avançados ao cliente, ou seja outros serviços que não sendo serviços de base interagem com estes para proporcionar aos clientes algumas facilidades úteis e atractivas. Os AS fornecem os mais variados serviços avançados desde conferências utilizando como suporte o MRFC, encaminhamento de chamadas, bloqueio de chamadas, serviço de indicação do estado do cliente (*Presence)*, etc.

O AS é contactado pelo S-CSCF. Os AS não conseguem encaminhar as mensagens para fora da rede o que significa que têm necessariamente de enviar as mensagens pelo S-CSCF que os contactou. Existem três tipos de AS: os que funcionam como ponto terminal da chamada, os que funcionam apenas como um ponto de transporte na rede e ainda os *Back to Back User Agent* (B2BUA)[3GPP TS 23.218 V6.4.0] que podem interligar duas chamadas diferentes. Um B2BUA funciona como ponto terminal no sentido *chamador* – B2BUA e funcionam como ponto de início de chamada no sentido

B2BUA – chamado. O IM-SSF é um AS e possibilita que o controlo e gestão de sessões multimédia IP sejam feitos através da lógica de um operador GSM. O IM-SSF é do tipo B2BUA porque de acordo com a lógica de serviço pode necessitar de estabelecer uma nova sessão para outro destino.

2.2 *IM-SSF*

Nesta secção é descrito sucintamente o comportamento do IM-SSF segundo as normas que o definem. Para isto é necessário explicar quais os protocolos que o IM-SSF utiliza para comunicar. Como AS, o IM-SSF deve conseguir comunicar em SIP e Diameter com a rede IMS. Do lado das redes inteligentes GSM deve conseguir comunicar em MAP e CAP que utilizam a pilha ***S**ignalling **S**ystem Number* **7** (SS7) para efectuar o transporte das suas mensagens.

2.2.1 Protocolos SS7

O SS7 é o responsável por transportar as mensagens entre o IM-SSF e a rede inteligente GSM.

O SS7 nasce como evolução do SS6 tornando-o mais adaptado ao diálogo entre sistemas digitais e conferindo-lhe mais flexibilidade para colmatar necessidades futuras.

O SS6 foi o primeiro sistema de sinalização de canal comum usado entre estações analógicas controladas por computadores.

A recomendação ITU Q.700 define o SS7 como um sistema de sinalização de canal comum com capacidade de transportar qualquer tipo de informação, no formato de mensagens, de forma fiável e na ordem correcta sem perda ou duplicação, optimizado para operar em redes de Telecomunicações com transmissão e comutação digitais. O SS7 é uma rede de pacotes que transporta mensagens entre utilizadores da rede que implementam vários tipos de funções. Para mais detalhes consultar [SS7 Teixeira de Sousa].

A Figura 2-4 apresenta os grupos funcionais que formam o sistema SS7.

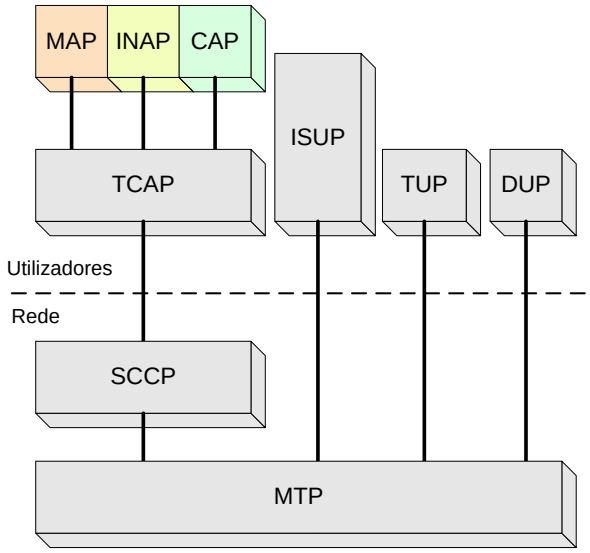

Figura 2-4: Pilha protocolar SS7

A estrutura do sistema SS7 pode ser vista globalmente como uma rede de sinalização controlada pelo *Message Transfer Part* (MTP), através da qual vários pares de utilizadores da rede trocam mensagens.

O MTP tem como função efectuar a transferência fiável das mensagens de sinalização entre utilizadores da rede de sinalização que comunicam entre si.

O *ISDN User Part* (ISUP) permite que os pares de módulos ISUP situados em estações telefónicas digitais diferentes troquem mensagens para o estabelecimento, monitorização e transmissão das chamadas telefónicas.

O *Telephone User Part* (TUP) é um procedimento para o estabelecimento de chamadas telefónicas na rede telefónica não *Rede Digital com Integração de Serviços* (RDIS) implementando funcionalidades semelhantes às sinalizações telefónicas anteriores ao SS7. O TUP é actualmente pouco utilizado tendo sido substituído por uma versão simplificada do ISUP.

O *Data Users Part* (DUP) fornece sinalização para o controlo de chamadas relacionadas com os serviços de transmissão de dados através de circuitos comutados. Tal como o TUP, foi pouco implementado.

O *Signalling Connection Control Part* (SCCP) acrescenta facilidades de endereçamento ao MTP indispensáveis no funcionamento das redes móveis. Inclui funções de gestão de

bases de dados situadas na rede de sinalização e do acesso a estas bases de dados de forma transparente às aplicações que as usam. Controla a disponibilidade dos seus utilizadores, os chamados subsistemas e difunde o estado de disponibilidade dos subsistemas aos outros subsistemas interlocutores. O SCCP permite disponibilizar serviços de rede *connectionless* e *connection oriented* via rede de sinalização para transferência de informação.

O ***T**ransaction **C**apabilities **A**pplication **P**art* (TCAP) é um *Application Part* que utiliza os serviços de SCCP. As TCAP endereçam a formatação dos dados e a sua apresentação em formatos normalizados através dos quais são invocadas remotamente operações via SCCP e rede de sinalização. Veiculam também a resposta às operações invocadas. Um exemplo é a tradução de um número verde numa base de dados remota pedida por uma estação telefónica. Os serviços das TCAP são utilizados por outras *Application Part,* como o MAP que é utilizado no ambiente GSM para consultar bases de dados tendo em vista a localização de um telefone móvel na rede GSM e o encaminhamento das chamadas de e para um telefone móvel.

O ***I**ntelligent **N**etwork **A**pplication **P**art* (INAP) é um *Application Part* (Recomendação ITU Q.1218) e é o protocolo usado, por exemplo, entre uma estação telefónica, capaz de detectar durante o processamento de uma chamada a presença de um pedido de acesso a um serviço de rede inteligente e uma aplicação que faz o controlo desse serviço de rede inteligente, o ***S**ervice **C**ontrol **P**oint*.

2.2.1.1 MAP

O ***M**obile **A**pplication **P**art* (MAP) define os protocolos e procedimentos de comunicação entre entidades da rede GSM. A sua especificação técnica é descrita em GSM 09.02: *Mobile Application Part specification.*

O MAP utiliza serviços disponibilizados pela pilha ***S**ignalling **S**ystem Number 7* (SS7). A pilha SS7 apresentada na Figura 2-4 é comparável ao modelo OSI. Comparativamente, o MAP insere-se ao nível das aplicações. As mensagens MAP têm uma parte transaccional

fornecida pelo *Transaction Capabilities Application Part* (TCAP) e uma parte operacional referente à mensagem a ser transportada, no caso mensagens MAP.

Como o MAP se baseia na pilha SS7, é necessário compreender um pouco, o que é o SS7 e o que fazem os seus componentes.

O MAP, no contexto desta dissertação, é utilizado para transportar informação específica do utilizador, entre o IM-SSF e o HSS. O protocolo MAP define um número significativo de mensagens mas, no contexto desta dissertação, apenas são necessárias as mensagens especificadas para o IMS mais precisamente para a interacção IM-SSF < – > HSS. Para melhor compreensão da interacção entre o IM-SSF e o HSS é apresentado na Figura 2-5 um exemplo demonstrativo.

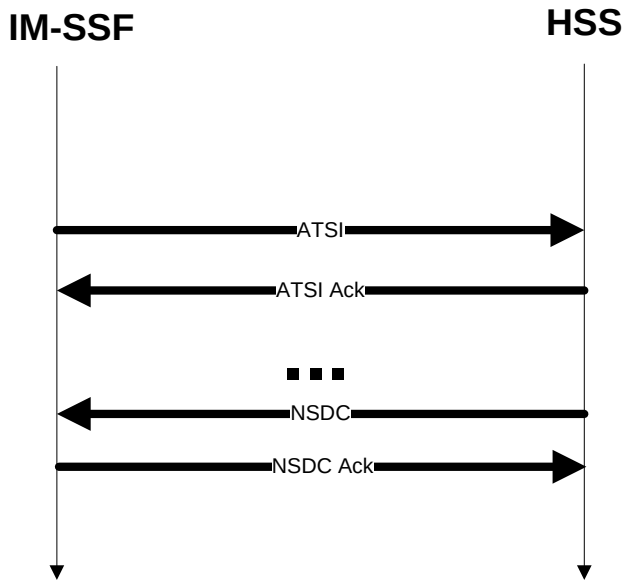

Figura 2-5: Fluxo MAP

O IM-SSF ao receber um registo de um cliente, envia o pedido *Any Time Subscription Interrogation request* (ATSI) para o HSS, pedindo a informação *IP Multimedia CAMEL Subscription Information* (IM-CSI) do cliente. O HSS responde com a informação pretendida na mensagem ATSI *ack*. Caso a informação do cliente registado seja modificada no HSS este envia a mensagem *Notify Subscriber Data Change* para o IM-SSF com a informação IM-CSI que sofreu alteração. O IM-SSF ao receber esta mensagem responde com a mensagem NSDC ack, indicando que a informação do

cliente foi actualizada. Este protocolo não foi muito aprofundado uma vez que se optou por não o implementar.

2.2.1.2 CAP

O ***C****ustomized Application for Mobile Network Enhanced Logic* ***A****pplication* ***P****art* (CAP) definido em 3GPP 29.078 é o protocolo usado para estabelecer o contacto entre o IM-SSF e o *gsmSCF*.

O ***C****ustomized* ***A****pplication for* ***M****obile Network* ***E****nhanced* ***L****ogic* (CAMEL) é uma iniciativa do 3GPP para efectuar a convergência das redes inteligentes na arquitectura das redes GSM fazendo uso de entidades e protocolos existentes em ambas. O CAMEL especifica os fluxos de informação que podem ser passados entre duas redes.

O CAP é uma evolução do protocolo INAP sendo o CAP mais restrito e menos flexível logo tem a vantagem de ser usado de uma forma mais global.

O CAP, à semelhança do MAP, utiliza serviços disponibilizados pela pilha SS7 para a construção do cabeçalho da mensagem. O corpo da mensagem é definido pelo CAP. No contexto deste documento, apenas são relevantes as mensagens CAP 4 IMS que na realidade são uma extensão às mensagens CAP 3, com a finalidade de incluir campos específicos do IMS.

No contexto desta dissertação, as mensagens relevantes do CAP enviadas pelo IM-SSF são:

- Initial Detection Point (IDP)
- Apply Charging Report
- Event Report BCSM
- Activity Test Ack
- Call Gap
- Call Information Report

As mensagens recebidas pelo IM-SSF no contexto CAP são:

- Connect

- Continue
- Apply Charging
- Request Report BCSM Event
- Cancel
- Release Call
- Connect To Resource
- Disconnect Forward Connection
- Continue With Argument
- Disconnect Forward Connection
- Call Information Request
- Furnish Charging Information
- Reset Timer
- Play Announcement
- Prompt And Collect User Information
- Specialized Resource Report

Para se compreender melhor como estas mensagens interagem e para que servem é dado um fluxo de informação demonstrativo de uma sessão CAP que permite o controlo de uma sessão e simultaneamente a taxação online do assinante.

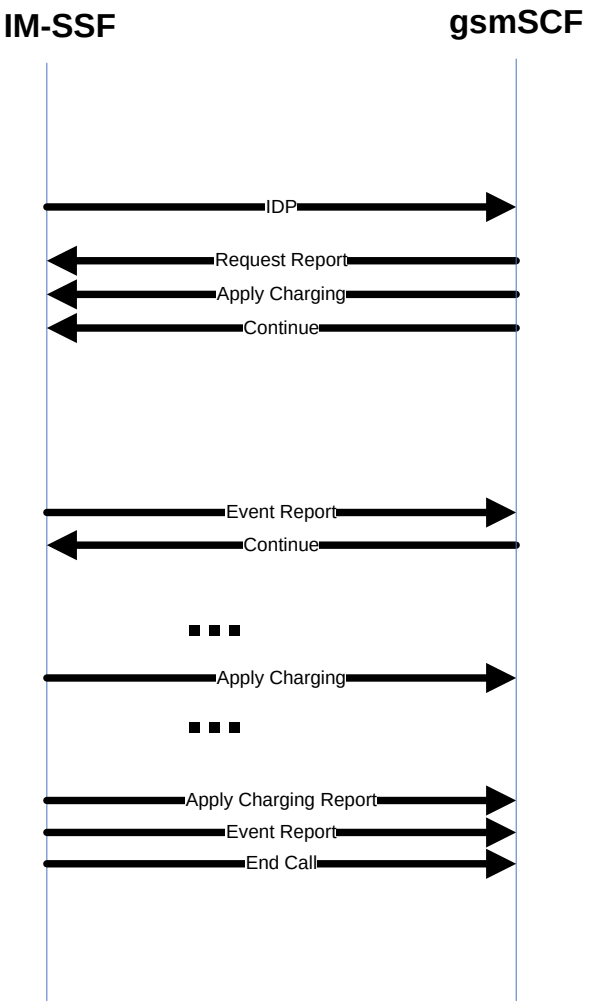

Figura 2-6: Fluxo CAP

A Figura 2-6 confirma que para iniciar um fluxo CAP é utilizada a mensagem IDP que transporta a informação relativa ao chamador, ao chamado, ao *gsmSCF* contactado e ao serviço a ser invocado. O *gsmSCF* analisa a informação recolhida na IDP e decide fornecer o serviço. Antes de conceder o serviço, o *gsmSCF* decide activar alguns pontos de detecção na chamada. Estes pontos de detecção ou em inglês ***Detection Points*** (DP), são pontos específicos que ocorrem numa chamada de voz (por exemplo: o estabelecimento da sessão, rejeição da sessão, cancelamento da sessão, etc...). A mensagem "Request Report BCSM Event" contém uma lista de DP que se pretendem activos nessa sessão.

Após o envio de activação dos pontos de detecção o *gsmSCF* decide que o serviço em causa necessita de informação de taxação em tempo real. Para isto é enviada a

mensagem "Apply Charging", que é uma indicação ao IM-SSF para iniciar os procedimentos necessários para guardar a informação desta sessão e reportá-la ao *gsmSCF*. No caso do IM-SSF, a informação da sessão relevante que será passada ao *gsmSCF* é o tempo de duração de uma sessão. O "Apply Charging" contém um campo que indica o intervalo de tempo fornecido ao cliente. Este intervalo funciona como uma espécie de saldo temporal findo o qual o IM-SSF envia um "Apply Charging Report" indicando o tempo total decorrido desde o estabelecimento da sessão. Após a activação da taxação o *gsmSCF* envia a mensagem "Continue" que indica ao IM-SSF que deve deixar a sessão prosseguir para o destino. Neste ponto o *gsmSCF* também poderia ter respondido com a mensagem "Connect" ou "Continue With Arguments". O "Connect" é usado para mudar o destino da sessão enquanto o "Continue With Arguments" é usado para modificar alguns parâmetros da sessão mas mantendo o destino. Após este processo, a chamada prossegue normalmente. Quando a sessão é estabelecida, é detectada a ocorrência do ponto de detecção armado previamente e é indicado ao *gsmSCF* esta ocorrência usando um "Event Report BCSM". Esta mensagem pode interromper a sessão ou não dependendo do modo como o ponto de detecção foi activado. Caso o ponto de detecção tenha sido activado como *interrupt* o IM-SSF quando ocorrer esse ponto de detecção envia um "Event Report BCSM" e aguarda que o *gsmSCF* forneça mais instruções. Caso o ponto de detecção tenha sido activado como *notify*, o IM-SSF, quando esse ponto de detecção ocorre, envia o "Event Report BCSM" e continua o processamento da sessão, não necessitando de instruções do *gsmSCF* para prosseguir. Quando a sessão termina é enviado um "Apply Charging Report" que indica o tempo decorrido desde o estabelecimento da sessão e é reportado o evento de terminação de sessão ao *gsmSCF*. No exemplo apresentado, o último evento é do tipo *notify*. Após o envio destas duas mensagens o IM-SSF indica que este fluxo CAP deve terminar. A indicação de que a sessão deve ser terminada é feita pela parte transaccional da mensagem que contem o estado da ligação. Desta forma o IM-SSF pode indicar que a

sessão vai terminar no último "Event Report BCSM" que envia ou então enviar uma mensagem só com a parte transaccional indicando o término da sessão (TC_END).

2.2.2 Módulo IM-SSF

O módulo *IP Multimedia Service Switch Functions* (IM-SSF), definido em [3GPP TS 23.278 V6.1.0], pode ser visto como um servidor de aplicações SIP que implementa a porta de ligação entre o protocolo SIP da interface *IMS Service Control* (ISC) (interface SIP designada pelo 3GPP para os AS, como padrão para a comunicação com a rede IMS) e o protocolo CAP que é utilizado na interface entre o IM-SSF e as funções de controlo de serviço GSM implementadas pelo módulo *gsmSCF*.

Através do IM-SSF, o S-CSCF passa o controlo de uma chamada multimédia IP, originada e/ou terminada numa estação móvel, para as funções de controlo de serviço (*gsmSCF*) tendo em conta a informação sobre o utilizador enviada previamente do HSS ao IM-SSF via interface MAP.

O módulo *gsmSCF*, tal como o nome indica, é um módulo que controla o serviço na rede GSM e cuja comunicação de sinalização da chamada é efectuada via o protocolo CAP sobre o TCAP da pilha SS7. Este é o ponto de acesso que o IM-SSF utiliza para fornecer os mesmos serviços avançados existentes no GSM às redes da nova geração baseadas no SIP. Com isto, começa-se a visualizar a grande vantagem do IM-SSF, pois possibilita aos operadores de redes móveis entrar nas redes da nova geração, reutilizando e rentabilizando, todo o material que já possuem e disponibilizando nesta nova rede, os mesmos serviços avançados que já forneciam aos seus clientes, através da rede GSM.

Na Figura 2-7 estão representadas as interfaces existentes no IM-SSF e quais os módulos que as irão usar.

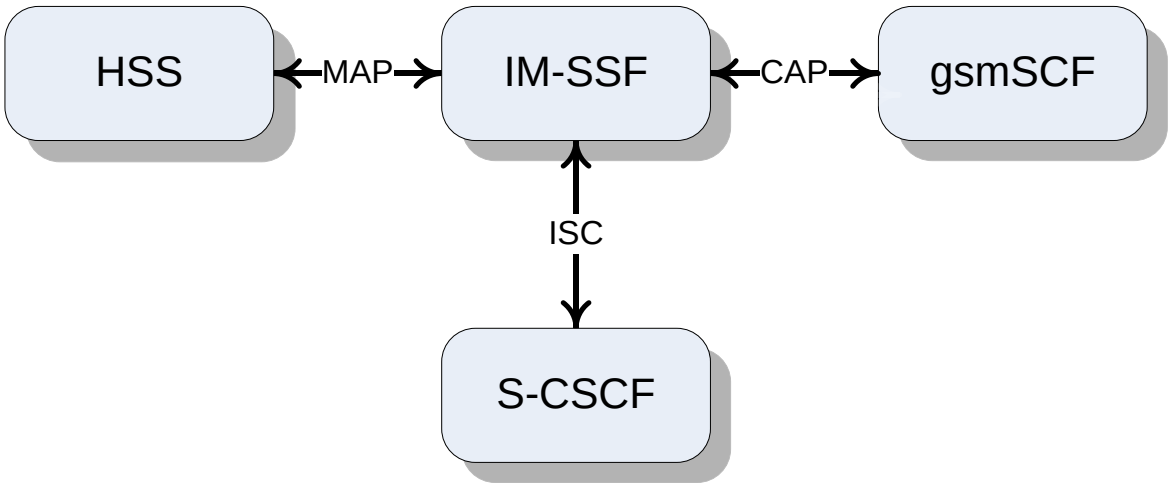

Figura 2-7: Modelo de interfaces do IM-SSF

Como foi explicado na secção 2.2.1.1 o protocolo MAP é utilizado pelo IM-SSF para possibilitar a comunicação com o HSS e deste receber os ***C****AMEL* ***S****ubscription* ***I****nformation* (CSI). Os CSI contêm informação específica do cliente e entre esta informação os serviços avançados subscritos por este. Com base no CSI recolhido, o IM-SSF decide se o serviço invocado pelo cliente é autorizado ou não. Depois da decisão e caso o serviço seja autorizado, o IM-SSF comunica com o *gsmSCF* que fornece a indicação de como deve ser tratada a sessão.

Na interface ISC é usado o protocolo SIP. É por esta interface que chega a informação transmitida pelo cliente. Ao receber um pedido na interface ISC, o IM-SSF efectua os seguintes passos:

- Verifica a existência dessa sessão
- Consulta o CSI do cliente
- Consulta o *gsmSCF* para receber instruções.

A norma [3GPP TS 23.278 V6.1.0] estabelece uma arquitectura para o módulo IM-SSF onde se faz claramente a divisão entre a parte do módulo que trata das mensagens chegadas pela interface ISC e da parte que estabelece relações de controlo com o *gsmSC*F. Com esta referência, o levantamento de requisitos e posterior desenvolvimento baseiam-se fundamentalmente nesta norma que define os

procedimentos que o módulo deve efectuar. Desta forma, o motor do módulo é uma representação fiel desta norma facilitando assim a detecção e correcção de erros e deixando liberdade para que o módulo possa crescer com o desenvolvimento da norma.

Existem dois cenários práticos de utilização do IM-SSF, *Mobile Originator* (MO) e *Mobile Terminator* (MT). Para perceber estes termos, basta compreender que os cenários MO dizem sempre respeito ao originador da sessão, e o cenário MT diz respeito ao destino da sessão. Considere-se, por exemplo, o toque de uma música ao chamador enquanto este espera que o destino atenda. Este tipo de serviço é o destino quem o subscreve e só quando se tenta estabelecer a sessão com esse destino é que uma determinada música (escolhida pelo destino) é tocada. Assim este é um serviço MT. Quem decide a passagem do estado MO para o estado MT é o S-CSCF. Por exemplo, se o Cliente A da rede IMS A estiver a efectuar uma chamada para o Cliente B da rede IMS B e o IM-SSF for contactado pela rede IMS A, a sessão é considerado no estado MO pois ainda se encontra no sentido ascendente. Por outro lado, se o IM-SSF for contactado pela rede IMS B a sessão é considerada no estado MT, pois já se encontra no sentido descendente.

Ao serviço MO são associados os CSI O_IM_CSI e D_IM_CSI. O O_IM_CSI é informação relativa ao originador, sendo o serviço fornecido à origem enquanto o D_IM_CSI é informação relativa ao originador tendo como base de análise o destino marcado pelo originador, ou seja o serviço é fornecido ao destino mas no entanto a sessão encontra-se ainda no lado originado (MO).

O VT_IM_CSI é o CSI associado ao serviço MT e contém informação relativa ao destino da chamada. Neste tipo de serviços quem requer o serviço é o destino e é com base neste que o serviço é fornecido.

O IM-SSF como AS do tipo B2BUA consegue funcionar como elemento SIP terminal ou como elemento SIP originador numa sessão. Este é o comportamento adoptado pelo IM-SSF, ou seja ao receber o pedido de estabelecimento de sessão (“INVITE”), após o processamento adequado, cria um novo pedido de estabelecimento de sessão para o

destino. Desta forma o IM-SSF possui dois ramos SIP (duas sessões SIP correlacionadas no IM-SSF), um ramo desde o chamador ao IM-SSF e outro ramo desde o IM-SSF ao chamado.

2.3 *Exemplo Ilustrativo*

Para melhor compreender a função do IM-SSF na rede IMS considere-se um exemplo de um assinante Antonio@ptinovação.pt que tenta efectuar uma chamada de voz com o assinante Bernardo@meditel.ma. A Figura 2-8 apresenta o fluxo de mensagens no IM-SSF correspondente ao exemplo apresentado.

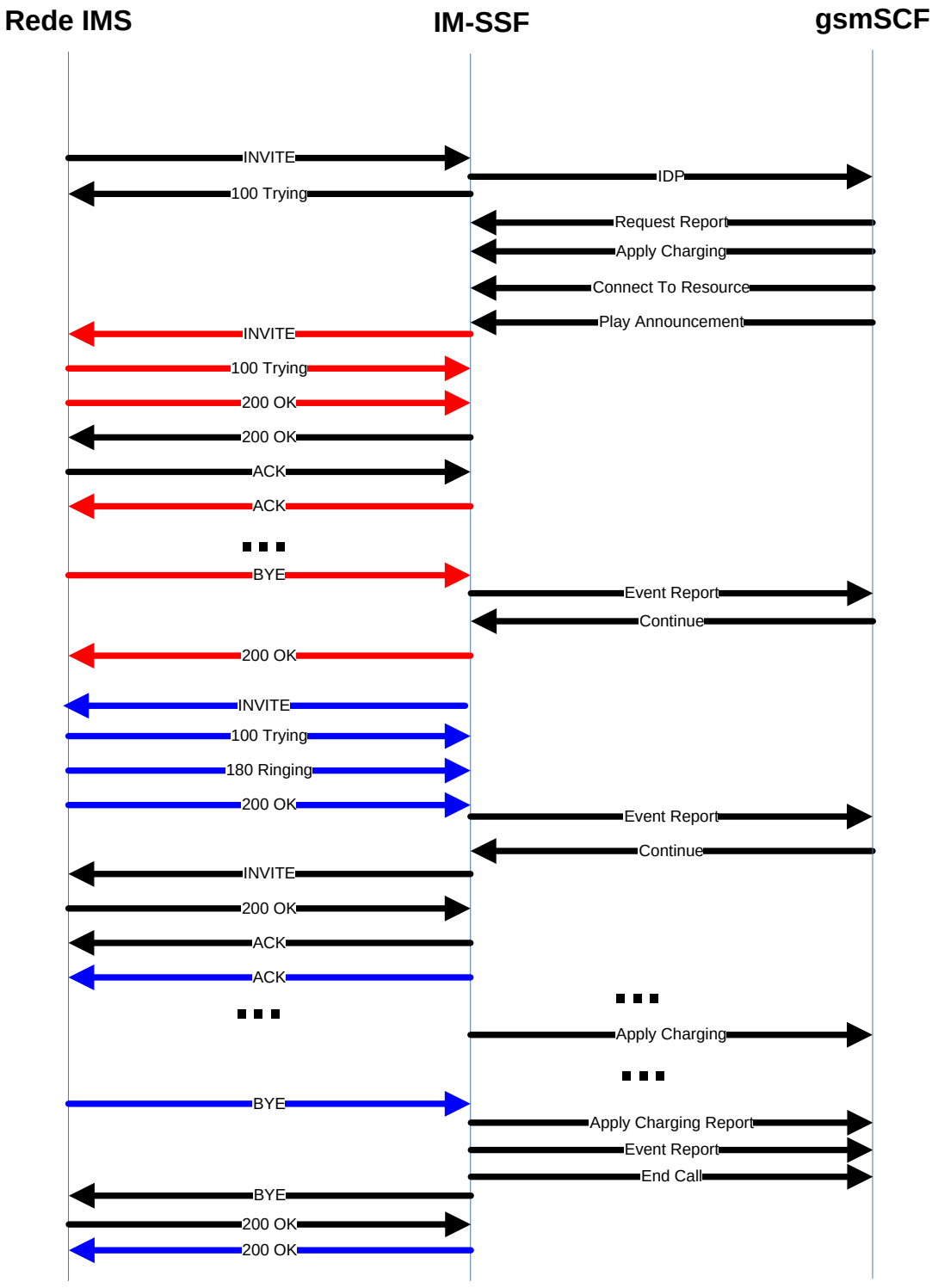

Figura 2-8: Fluxo de mensagens no IM-SSF

あ António, depois de registado na rede IMS, pretende efectuar uma chamada para Bernardo. Após o registo, o equipamento do chamador já conhece o caminho para chegar ao S-CSCF que o serve. Assim, o equipamento envia um "INVITE"

que passa pelo P-CSCF e S-CSCF utilizados no registo. O facto do António se encontrar em *roaming* ou não, apenas é distinguido pelo P-CSCF ser ou não ser, pertencente à rede do mesmo (ptinovação.pt).

- Quando o "INVITE" chega ao S-CSCF, este analisa os IFCs de António recolhidos no processo de registo. Caso estes não tenham sido recolhidos na fase de registo, o S-CSCF pode ainda consultar o HSS via Diameter para os obter. Através da análise dos IFCs, o S-CSCF sabe se existe algum serviço a activar quando o António tenta estabelecer uma chamada. Assumindo a existência dum serviço providenciado por uma AS (por exemplo, o IM-SSF), o S-CSCF reencaminha o "INVITE" para esse AS.
- Ao receber o "INVITE", o IM-SSF contacta o *gsmSCF* para obter instruções. O serviço aqui descrito é o toque de um anúncio indicando a António que a sua chamada está a ser encaminhada.
- O IM-SSF envia a IDP para o *gsmSCF*. O *gsmSCF* responde com: um "Request Report BCSM Event", indicando os pontos na sessão dos quais pretende ser informado, um "Apply Charging", um "Connect to Resource" e um "Play Announcement". As duas últimas mensagens indicam ao IM-SSF que deve contactar um recurso especializado (MRFC) e qual o anúncio que deve ser requerido. Com esta informação o IM-SSF envia o"INVITE" para o S-CSCF com destino o MRFC. Este "INVITE"contém a informação necessária ao MRFC para proceder ao toque do anúncio.
- O S-CSCF encaminha o pedido para o MRFC.
- Ao receber o pedido de "INVITE", o MRFC processa-o e responde com um "200 OK", que é encaminhado pelo S-CSCF para o AS.
- O IM-SSF envia um "200 OK" para o equipamento de António. Este "200 OK" é encaminhado pelo percurso inverso percorrido pelo "INVITE" ou seja S-CSCF, P-CSCF e equipamento do António.

- Ao receber o "200 OK", o equipamento de António envia um "ACK" e fica pronto para receber a mensagem de voz.
- O "ACK" é encaminhado para o IM-SSF via P-CSCF e S-CSCF.
- O IM-SSF envia um "ACK" para o MRFC via S-CSCF. Neste ponto a ligação está estabelecida e é tocado o anúncio pelo MRFC.
- Quando o anúncio termina, o MRFC envia um "BYE" para o AS via S-CSCF que responde com "200 OK". Considerando o AS, o IM-SSF, é contactado mais uma vez o *gsmSCF* que indica que a chamada deve prosseguir para o destino, o Bernardo. Neste ponto o IM-SSF envia um "INVITE" para o S-CSCF.
- O S-CSCF analisa os IFCs para chamadas provenientes de António e verifica que não deve enviar o pedido para mais nenhum AS. Então, o S-CSCF passa a funcionar em modo terminado ou seja, analisa o destino para saber para onde deve encaminhar o "INVITE". Após análise do destino, o S-CSCF verifica que este se encontra na rede *meditel.ma* que é servida por um determinado I-CSCF dessa mesma rede. Assim, o S-CSCF envia o "INVITE" para o I-CSCF da rede acima referida.
- O I-CSCF analisa o destino da mensagem e contacta o HSS num processo semelhante ao de registo, para obter o S-CSCF que serve o assinante para quem se dirige a mensagem. Após descoberta do S-CSCF que serve o assinante destinatário o "INVITE" é encaminhado para esse S-CSCF.
- O S-CSCF verifica se tem algum serviço que seja activado em modo terminado para o destinatário. O modo terminado é aqui despoletado directamente pois o S-CSCF verifica que o assinante originador não é servido por si. Após esta verificação e caso não ocorra nenhum serviço, a sessão é encaminhada para o P-CSCF e por sua vez para o equipamento de Bernardo.
- O equipamento de Bernardo começa a tocar e informa a rede deste facto através da mensagem "180 Ringing". Esta mensagem percorre o caminho até ao AS ou

seja P-CSCF, S-CSCF, I-CSCF da rede *meditel.ma*, e S-CSCF, AS da rede *ptinovação.pt*. O IM-SSF não passa esta informação ao terminal do António pois para este a sessão já se encontra estabelecida e o "180 Ringing" no contexto SIP não faz sentido após a sessão estabelecida.

- Quando Bernardo atende, envia um "200 OK" que percorre o mesmo percurso que percorreu o "180 Ringing".
- O IM-SSF, que neste exemplo está a fornecer um serviço de MO ao António, envia uma mensagem de "UPDATE" para António a informar que agora deve efectuar a ligação de voz com Bernardo.
- O equipamento do António responde com um "ACK" que chega ao AS que por sua vez envia um "ACK" ao Bernardo como resposta ao seu "200 OK". A partir deste momento a ligação está estabelecida.
- No final, Bernardo desliga a chamada enviando um "BYE". Este "BYE" é encaminhado pela rede do Bernardo até ao AS que por sua vez encaminha o "BYE" até ao António. O percurso efectuado pelo "BYE" na rede do Bernardo é o seguinte: Bernardo, P-CSCF, S-CSCF, S-CSCF da rede do António. Isto quer dizer que o I-CSCF já não faz parte do percurso da mensagem. Como o I-CSCF é um elemento apenas importante para o início da sessão, este opta por não se colocar no caminho de futuros pedidos, embora tenha sempre de receber as respostas para os pedidos que já tenha recebido.

Poder-se-ia questionar o porquê de após o "180 Ringing" ter passado pelo I-CSCF, o "200 OK" também passe pelo mesmo. Isto deve-se ao facto de se convencionar que as mensagens de resposta sejam encaminhadas pelos campos SIP VIA que indicam o caminho percorrido pela mensagem do tipo *request*. Assim só um novo *request* é que tem a capacidade de escolher um novo caminho. No término da sessão, ambos os intervenientes, recorrendo a outro campo SIP de roteamento, o *Record Route*, sabem qual o percurso pelo qual devem enviar as mensagens do tipo *request.* Como o I-CSCF não tem qualquer papel após o inicio da sessão, normalmente não inclui o seu endereço

SIP no campo *Record Route* e, assim, não recebe *requests* subsequentes para essa sessão, libertando-o para desempenhar outras tarefas mais importantes que o encaminhamento SIP. Esta é a razão pela qual o BYE final já não contacta o I-CSCF.

Capítulo 3 - Desenvolvimento

Na fase de desenvolvimento de um qualquer sistema, existem passos fundamentais a cumprir, componentes a desenvolver e estratégias a implementar. Este capítulo descreve todo o trabalho efectuado no decorrer do desenvolvimento do sistema IM-SSF. O trabalho de desenvolvimento de um sistema de software, segundo a norma UML, passa por várias fases distintas: a análise de requisitos, a construção de um modelo, a concepção, o desenvolvimento, teste do módulo e por fim a sua integração.

3.1 *Fundamentação e Abordagem, Restrições e Condicionantes*

A abordagem seguida para a definição da arquitectura do sistema baseou-se no paradigma de computação distribuída. Ao separar o processamento em diversas etapas distintas (e distribuídas por diferentes componentes) consegue-se aumentar o desempenho e modularizar a solução (ver secção Modularidade – Critérios e Regras).

Foi assumido como uma restrição de implementação que cada instância do componente IM-SSF apenas trata um *Basic Call State Model* (BCSM), pois na rede IMS quem decide qual o estado da sessão (originado ou terminado) é o S-CSCF que por sua vez invoca os AS (IM-SSF). Isto implica a existência de duas instâncias desse componente sempre que se pretenda tratar o *Originating/Dialing IP Multimedia BCSM (*O/D-IM-BCSM) e o *Terminating IP Multimedia BCSM* (T-IM-BCSM). De notar que o D-IM-BCSM é considerado como parte do O-IM-BCSM e não como independente. Isto deve-se ao

facto dos serviços de *dialing* só fazerem sentido para o Originador, sendo este o único que no estabelecimento da sessão consegue digitar os dígitos.

Para o desenvolvimento do software, foi escolhido utilizar a plataforma *Linux* e utilizando como linguagem de programação o C++ pois o projecto foi desenvolvido na PT Inovação que utiliza estes recursos com maior frequência, tendo já algumas bibliotecas C++ que facilitam o desenvolvimento do software, abstraindo o programador um pouco das especificidades do sistema operativo.

3.1.1 Modularidade – Critérios e Regras

A modularização correcta de sistemas proporciona, de uma forma geral, um sistema mais robusto, que lida melhor com eventuais situações de falha e onde os seus componentes são mais facilmente reutilizáveis.

Segundo [OO Meyer], um paradigma deve cumprir 5 critérios e 4 regras para ser modular.

Critérios:

C1: Desagregação: Um paradigma cumpre este critério se promover a decomposição do problema em problemas menores.

C2: Agregação: Pode-se construir um sistema complexo através da junção de vários componentes.

C3: Inteligibilidade modular: cada módulo é inteligível por si só, ou seja, as funcionalidades de cada módulo são referentes unicamente ao propósito para o qual o módulo é desenvolvido, i.e., o módulo pode ser integrado noutros sistemas.

C4: Continuidade modular: uma mudança na especificação apenas afecta um ou alguns (poucos) módulos do sistema. Isto evita que uma alteração na especificação implique uma reconstrução de todos os componentes do sistema.

C5: Protecção modular: a ocorrência de uma condição anormal na execução de um módulo fica confinada a esse módulo ou apenas se propaga para alguns (poucos)

módulos vizinhos. Desta forma qualquer problema com um determinado componente do sistema não se propaga a todo o sistema.

Regras:

R1: Mapeamento directo: isto implica que deve ser mantido um paralelo entre o modelo do sistema (documentação) e a implementação real do sistema. A aplicação desta regra deve seguir os critérios: agregação, inteligibilidade modular e continuidade modular.

R2: Poucas interfaces: cada módulo deve manter poucas interfaces para diminuir a complexidade e minimizar as dependências entre os módulos. A aplicação desta regra deve seguir os critérios: continuidade modular e protecção modular.

R3: Encapsulamento: cada módulo deve mostrar para o exterior apenas a informação estritamente necessária, reduzindo a complexidade e informação redundante. A aplicação desta regra deve seguir os critérios: continuidade modular e protecção modular

R4: Módulo aberto-fechado: um módulo deve ser suficientemente aberto para que possa ser extensível de forma a conseguir implementar facilmente novas funcionalidades. O módulo deve ser fechado para que se possa utilizá-lo, exportá-lo ou reutilizá-lo. Isto significa que o módulo deve estar terminado o suficiente para que possa ser usado mas flexível o suficiente para que se possam implementar novas funcionalidades.

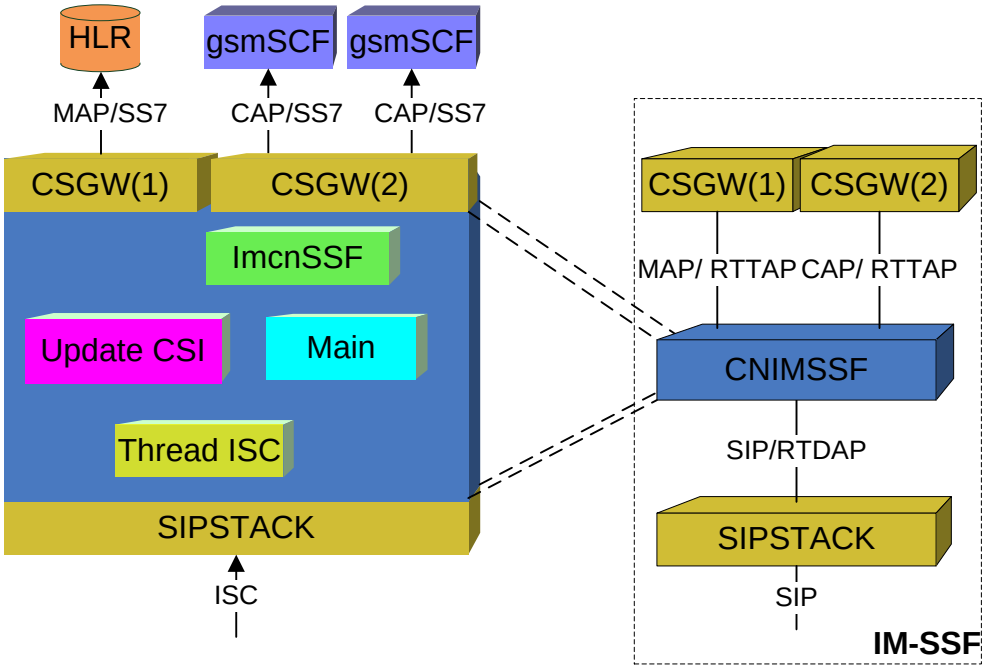

Figura 3-1: Sistema IM-SSF desenvolvido

Na Figura 3-1 é apresentada a arquitectura pensada para o sistema IM-SSF. Do lado direito da figura são apresentadas as quatro threads (ImcnSSF, Update CSI, Main e Thread ISC) que compõem o CNIMSSF e se encontram explicadas no capítulo 3.4.3 .

O sistema IM-SSF é composto por quatro (4) módulos (C1 Desagregação). O CNIMSSF é o módulo que detém a inteligência do sistema de acordo com o especificado na norma [3GPP TS 23.278 V6.1.0]. O CNIMSSF foi partido em quatro processos (aplicação do critério C1, Desagregação): o Thread ISC, o Update CSI, o ImcnSSF e o Main. A motivação para existir o CSGW (1) é permitir converter as mensagens MAP sobre RTTAP para MAP/SS7 ou então Diameter. O CSGW (2) permite o mapeamento de CAP/RTTAP para CAP/SS7, podendo contactar múltiplos *gsmSCF* de acordo com o serviço. Esta divisão permite que o CNIMSSF se encontre na rede IP deixando apenas um módulo (CSGW) interagir com a rede SS7 e toda a complexidade inerente. A SIPSTACK converte o SIP para SIP/RTDAP. Nesta conversão é apenas passada a informação necessária ao CNIMSSF para identificar os intervenientes e em que estado se encontra a sessão. Com isto, as mensagens trocadas entre a SIPSTACK e o CNIMSSF são mais pequenas do que as mensagens SIP, permitindo que o encaminhamento SIP e

outras informações apenas relevantes para o SIP, sejam armazenados apenas na SIPSTACK.

O RTDAP é um protocolo proprietário da PT Inovação que é suportado por algumas das soluções da PT Inovação. Sendo um elemento uniformizador das soluções permite a composição de novas soluções trocando apenas alguns módulos.

O RTTAP também é um protocolo proprietário da PT Inovação, no entanto foi desenhado com o intuito de transportar todo o conteúdo das mensagens CAP sobre IP para que a transposição do IP para o SS7 e vice-versa, seja directa.

O posicionamento dos módulos SIPSTACK, CNIMSSF e CSGW pode ser remoto possibilitando a distribuição do sistema IM-SSF por diferentes máquinas eventualmente com diferentes arquitecturas. Esta divisão permite que cada módulo seja inteligível por si só (critério C3 Inteligibilidade modular) e que uma alteração numa especificação apenas influencie um dos módulos (critério C4 Continuidade modular), por exemplo uma alteração nas mensagens SIP apenas afecta o SIPSTACK. O sistema IM-SSF foi construído recorrendo a alguns módulos já existentes na PT Inovação pelo que, desta forma, esta solução cumpre o critério C2 Agregação. O critério C5 Protecção modular, é cumprido pois caso o SIPSTACK ou o CSGW falhe, somente o CNIMSSF detecta e trata o problema não deixando transparecer o problema para os outros módulos. Poder-se-á pensar que caso o CNIMSSF falhe, todo o sistema falha pelo que não se cumpre o critério C5. Embora isto seja verdade, foi uma escolha consciente que tal acontecesse pois, sendo o CNIMSSF quem tem a inteligência do sistema, pretende-se que caso este falhe todos os outros módulos o saibam e possam actuar convenientemente (por exemplo, a SIPSTACK começa a rejeitar novas sessões SIP e termina as sessões em curso).

Ao cumprir todos os critérios pode-se afirmar que o sistema IM-SSF cumpre todas as regras visto que cada regra é apenas um agrupamento de alguns critérios. A Regra 5 aberto-fechado, é cumprida recorrendo ao versionamento do software: quando uma versão do software está suficientemente estável, ela pode ser utilizada; no entanto, é

possível efectuar novos desenvolvimentos, criando novas versões, sem contudo afectar os clientes que utilizam versões mais antigas.

3.1.2 Comunicação entre processos

Devido à escolha efectuada para a arquitectura do sistema (definida em 3.1.1) e à natureza do sistema, é necessário que os diferentes processos comuniquem entre si. A comunicação entre processos é fundamental em qualquer software, visto ser imprescindível comunicar os resultados produzidos. No IM-SSF isto é ainda mais imperioso pois, como se trata de um elemento de ligação entre duas redes (no caso do presente trabalho, a rede IP e a rede GSM), ele tem de constantemente comunicar com outros elementos das duas redes.

3.1.2.1 Mecanismos Possíveis

Os mecanismos de comunicação entre processos podem ser classificados nas seguintes categorias:

- *Pipes*
- *FIFOs*
- Memória partilhada
- Memória mapeada
- *Queues* de mensagens
- Semáforos
- *Sockets*

As descrições que se seguem estão em conformidade com o livro *Advance Linux Programming* [ALP Samuel].

Pipes

Os *pipes* providenciam um fluxo de comunicação unidireccional entre processos dentro do mesmo sistema operativo. Um *pipe* é criado através da execução da função do sistema operativo "*pipe(filedes)*" que cria um par de descritores de ficheiro devolvidos pelo argumento *filedes*. O *filedes[0]* é usado para leitura e o *filedes[1]* é usado para a

escrita. Isto significa que para se poder utilizar um *pipe* é necessário que os processos tenham alguém que lhes passe os descritores. Isto pode ser feito usando um processo que cria o *pipe* e depois cria os vários processos que utilizam o *pipe*, passando-lhes os descritores do *pipe*.

Os *pipes*, tal como o nome indica, podem ser vistos como canos onde numa extremidade é colocada informação e na outra é recebida informação. Nos *pipes*, a ordem de leitura é dada pela ordem de escrita dos dados e ao ler os dados estes deixam de existir no *pipe*. O acesso a um *pipe* é feito no Linux da mesma forma que o acesso a um ficheiro, contudo o ficheiro é puramente virtual (não existe fisicamente). Os *pipes* são normalmente bloqueantes, ou seja, quando um processo tenta ler de um *pipe* e caso não tenha informação para ler, o processo fica bloqueado. Da mesma forma, um processo que tente escrever num pipe que esteja cheio, fica bloqueado até que o outro processo retire dados para que este possa escrever. Os *pipes* duram apenas o ciclo de vida do processo que o criou.

FIFOs

Os FIFOs (*First In, First Out*) são semelhantes aos *pipes*, providenciando também um fluxo de dados *half-duplex*. A diferença em termos conceptuais entre um *FIFO* e um *pipe* é que o *FIFO* é identificado pelo sistema com um nome e os *pipes* não, ou seja, os *FIFOs* são *pipes* com nomes. Assim, um *FIFO*, ao contrário de um *pipe,* pode durar para além do ciclo de vida do processo que o criou (um *FIFO* mantém a sua existência no sistema até ser explicitamente removido).

No Linux quando é criado um FIFO é criado um ficheiro com o nome do FIFO e é este ficheiro que é utilizado para guardar os dados do FIFO. O acesso ao FIFO para escrita ou leitura é semelhante ao de um ficheiro.

Memória partilhada

Este mecanismo permite que diferentes processos comuniquem como se partilhassem um espaço de endereçamento virtual (zona de memória que não necessita de ser contigua). Qualquer processo que partilhe esta memória virtual pode lá escrever ou ler.

Este mecanismo reserva uma parte da memória para ser utilizada por diferentes processos. Tal como os *pipes*, é necessário existir um processo que disponibilize esta memória virtual aos outros processos, o processo pai.

Este é o mecanismo mais rápido nos sistemas de comunicação entre processos. Contudo, não garante a consistência dos dados, pois existe a possibilidade de dois processos acederem à memória ao mesmo tempo possibilitando a escrita e leitura simultânea, criando assim conflitos. Por esta razão, é imperioso que o acesso à memória seja mutuamente exclusivo.

Memória mapeada

A memória mapeada é uma memória partilhada mas com um nome. Da mesma forma que os FIFOS estão para os *pipes*, a Memória mapeada está para a memoria partilhada. Este tipo de memória em Linux consiste em associar um ficheiro a uma zona de memória. Desta forma sempre que se escreve ou lê no ficheiro, está-se na realidade a escrever ou a ler na memória. Neste tipo de comunicação de processos, é necessário um mecanismo que regularmente efectue uma sincronização entre o conteúdo da memória e o ficheiro. No Linux é o sistema operativo que se responsabiliza por esta tarefa.

Considere-se, por exemplo, um ficheiro que contém dados específicos num formato conhecido pelo utilizador (o ficheiro pode ser necessário para a execução de alguma aplicação). Neste caso, tem de existir uma função que procure dados no ficheiro e eventualmente os devolva. A função em causa tem de aceder sempre ao ficheiro de cada vez que é invocada, pelo que se for invocada muitas vezes numa aplicação, esta torna-se consideravelmente lenta devido aos acessos ao disco duro *(hard drive*). Colocando os dados do ficheiro em memória virtual, consegue-se uma melhoria significativa nos tempos de acesso aos dados pois o acesso à memória é muito mais rápido do que o acesso ao disco.

A técnica de mapeamento em memória consegue diminuir os tempos de computação pois o ficheiro é associado logicamente a uma porção da memória virtual ou ao espaço

de endereçamento do programa e com isto todas as operações passam a ser simples acessos à memória.

O *Linux* providencia as funções de sistema *mmap* (para instanciar) e *munmap* (para libertar) com o propósito de usar este mecanismo, que providencia a partilha de ficheiros mapeados. É necessário associar o ficheiro de tal maneira que esta associação se torna acessível a qualquer processo. Novamente podem surgir problemas de sincronização visto que podemos ter vários processos a competir pelo mesmo recurso.

Queues de mensagens

Este mecanismo possibilita o envio de informação entre processos de uma forma assíncrona. Assíncrono, neste caso, significa que quem enviou não precisa de estar à espera de notificação do receptor ficando desta forma liberto para continuar o processamento. Do outro lado, o receptor também não fica à espera caso não existam mensagens na *queue*. Neste tipo de comunicação, o tamanho das *queues* é configurável e é possível ter mais do que uma mensagem na *queue*. Isto faz das *queues* uma espécie de área de armazenamento onde podem ser armazenadas mensagens enquanto se efectua o processamento de outras mensagens. Este comportamento melhora a eficiência do processo receptor e a gestão de recursos da máquina.

No contexto desta dissertação, as *queues* são criadas no espaço de endereçamento do módulo e são partilhadas da mesma forma que os pipes seriam, por uma espécie de herança.

Numa situação de excesso de carga, como os recursos da máquina são finitos existe a possibilidade das *queues* ficarem cheias. Quando uma fila de espera do tipo FIFO atinge o limite, existem vários procedimentos que se podem efectuar para minimizar o atraso global das mensagens, que num software de tempo real é o factor mais importante.

De seguida são apresentados alguns destes procedimentos e as suas características mais importantes:

- **Drop–From–Tail.** Como o nome indica, este procedimento remove a última mensagem que chegou à fila. Por outras palavras, quando o sistema tenta colocar o pacote na fila e esta está cheia, então o pacote é descartado.
- **Drop-from-head.** Neste método, quando o sistema tenta colocar uma mensagem na *queue* e esta está cheia, o primeiro pacote da fila (o mais antigo, e próximo a ser processado) é descartado e é inserido no fim da *queue* a nova mensagem que o sistema tinha acabado de recolher. Este é um bom método para *Real Time* pois descarta as mensagens cujo tempo no sistema é o mais elevado. Isto não é muito grave em sistemas de Real Time pois normalmente as mensagens são retransmitidas ao fim de algum tempo ou o serviço é rejeitado por expiração de um temporizador.
- **Random drop.** Neste método é escolhido aleatoriamente uma mensagem de entre as que estão na *queue* e esta é removida. A mensagem nova que chegou é inserida no fim da *queue*.
- **Flush queue.** Neste método, o sistema ao detectar a *queue* cheia esvazia-a para poder colocar as novas mensagens na *queue*. Este é um método simples de implementar e acaba com a congestão dando algum tempo ao sistema para recuperar até a fila encher novamente.
- **Intelligent drop.** Este método descarta mensagens da fila de acordo com o conteúdo de cada mensagem. Todos os métodos apresentados até aqui são mais simples de implementar, no entanto são injustos para sessões que contenham maior número de mensagens na fila, visto serem métodos completamente agnósticos a contextos de sessão. O *intelligent drop* é o método mais justo de todos e como é feita a análise por conteúdo é possível minorar o impacto do congestionamento não removendo as mensagens de determinadas sessões. O facto de ser efectuada análise do conteúdo pode ser uma desvantagem pois pode ser necessário descodificar muita informação da mensagem para ser tomada uma decisão, tornando-o um método mais lento do que os anteriores.

Semáforos

Os semáforos podem ser considerados um mecanismo de comunicação entre processos pois permitem que dois processos comuniquem entre si para chegarem a um acordo de qual dos dois processos deve aceder ao recurso que ambos pretendem. No entanto, os semáforos por si só, não efectuam a troca de dados inerente às outras formas de comunicação entre processos. Contudo, pela simplicidade e história dos semáforos, estes são considerados suficientemente relevantes para serem aqui apresentados.

Os semáforos são um mecanismo de sincronização, que utiliza o conceito conhecido nas redes rodoviárias como semáforo, ou seja, quando um semáforo se encontra verde é indicação que o cruzamento se encontra disponível para avançar e quando o semáforo se encontra vermelho é indicação que o cruzamento se encontra ocupado e deve-se aguardar. Na comunicação entre processos, aplica-se exactamente o mesmo conceito; um processo tenta aceder a uma zona partilhada e coloca o semáforo a vermelho, um processo que queira aceder à mesma zona aguarda que o sinal fique verde para aceder. Quando o processo que se encontra na zona partilhada termina o seu processamento nessa zona, coloca o semáforo a verde indicando a qualquer processo que se encontre em espera que a zona partilhada se encontra acessível. Este é o conceito básico que permite uma sincronização dos processos a uma zona de dados partilhada.

Sockets

Os *sockets* são o mecanismo mais popular de entre todos os mecanismos de comunicação entre processos pela simples razão de permitirem, ao contrário dos anteriores, a comunicação entre processos executados em diferentes máquinas. Os *sockets* providenciam um canal de comunicação *full-duplex* entre processos que não necessitam de estar na mesma máquina.

Um *socket* é basicamente um ponto de acesso tornado acessível ao processo que o criou. O *socket* agrupa os dados em grupos a que chamamos pacotes, sendo estes transmitidos pelo *socket*.

Quando se cria um *socket*, é necessário especificar três parâmetros: o domínio, o tipo e o protocolo.

O domínio define a família do protocolo que será utilizado na comunicação, por exemplo IPv4, IPv6, ITU-T X.25, etc.

O tipo especifica a semântica da ligação, i.e., especifica que a ligação é do tipo *datagrama*, *stream*, *datagrama* com garantia de entrega, etc. Por exemplo com o domínio IP o tipo pode ser *stream* (usa o protocolo TCP), ou *datagram* (usa o protocolo UDP). Os pacotes ao circularem pela rede IP podem passar por vários equipamentos de encaminhamento que podem enviar alguns pacotes por um caminho e outros por outro, desta forma é possível um pacote que foi enviado depois de outro chegar antes desse ao destino. No estilo *stream* é garantido que os pacotes são entregues ao destino pela ordem que foram enviados, ao passo que no estilo *datagram* isto não é garantido.

O protocolo distingue um protocolo específico a ser utilizado num socket. Podem existir situações onde existe mais do que um protocolo para um determinado domínio e tipo, no entanto o normal é só existir um determinado protocolo. Por exemplo com o domínio IPv4 e tipo *stream* existe o protocolo TCP, neste caso o parâmetro protocolo pode ser preenchido com o valor zero (0), que indica que deve ser utilizado o protocolo disponível.

Para ser possível a comunicação entre dois processos utilizando *sockets*, além de definir o socket é necessário definir também o endereço onde o outro processo se encontra para que a comunicação se estabeleça. A utilização de *sockets* para a comunicação entre processos pode ser feita segundo dois modelos: o modelo *cliente-servidor* e o modelo *peer to peer*.

Modelo cliente-servidor

Neste modelo de comunicação, o servidor instancia um *socket* num porto tornando-o público aos clientes. Neste *socket*, o servidor fica à escuta de pedidos vindos dos clientes. O cliente por seu lado instancia um *socket* para estabelecer a comunicação com o servidor (neste caso o porto do cliente não necessita de ser previamente conhecido

visto ser ele a iniciar a ligação). Com o *socket* criado, o cliente tenta criar um canal de comunicação com o servidor no porto previamente conhecido. Neste modelo de comunicação o cliente é quem inicia a comunicação.

Modelo Peer to Peer (P2P)

Ao contrário do modelo cliente-servidor, no modelo *Peer to Peer* não existe um servidor nem um cliente fixo e todos podem iniciar uma comunicação com os *peers* que conhecem (um *peer*, ao receber o pedido de ligação de um outro *peer*, pode aceitá-lo tal como faria um servidor; no entanto, este *peer* também pode iniciar uma ligação para outro *peer*). Neste tipo de modelo, um *peer* necessita de conhecer a informação de onde outro *peer* se encontra para se poder ligar. Em termos de *sockets*, isto implica que um *peer* tenha aberto um *socket* para escuta de ligações de outros *peers* e que por sua vez tenha a capacidade de abrir um *socket* para iniciar a ligação com outros *peers*. Do ponto de vista de programação, um modelo cliente-servidor é mais simples de implementar pois só é necessário criar um servidor ou um cliente por interface ao passo que no modelo *peer to peer* seria necessário implementar no mínimo um servidor (este servidor aceita ligações de todas as interfaces do sistema) e um cliente por cada interface.

3.1.2.2 Mecanismos Adoptados

Para a comunicação com outros processos (programas diferentes) foi escolhida a comunicação via *sockets*. Este tipo de comunicação foi escolhido visto ser o único capaz de estabelecer comunicação entre dois processos existentes em máquinas distintas.

Pela simplicidade de desenvolvimento, decidiu-se optar pela solução cliente-servidor para estabelecer a comunicação com outros módulos. No caso do presente trabalho, assumiu-se que o módulo CNIMSSF é o cliente em todas as suas interfaces. Desta forma consegue-se dividir claramente as várias interfaces tendo um cliente por cada interface. Com isto consegue-se uma maior mobilidade dos vários módulos que compõem o sistema, e potencialmente um aumento da eficiência do sistema colocando diferentes módulos em máquinas distintas. Esta solução permite também ter um sistema mais

tolerante a falhas bastando para isso colocar réplicas dos vários módulos em máquinas distintas. Desta forma, quando um módulo activo é desactivado por qualquer motivo, pode ser usado um módulo semelhante existente noutra máquina.

Após a análise das várias formas de comunicação entre processos, decidiu-se que para a comunicação entre *threads* (dentro do mesmo programa) a melhor solução seria utilizar as *queues* de mensagens e memória partilhada pelas razões que a seguir se enunciam.

A memória partilhada é usada para guardar os dados de sessão necessários para o funcionamento do módulo no sistema. Por exemplo, é necessário que a informação recebida na interface SIP (thread ISC do CNIMSSF) seja transmitida para a interface CAP (ImcnSSF do CNIMSSF), isto concretiza-se usando um *array* de estruturas que é criado pelo programa principal (main) e disponibilizado o seu acesso às várias threads do módulo. A consistência dos dados é garantida através da organização da estrutura deste *array* que no local onde uma *thread* escreve todas as outras só podem ler.

As *queues* de mensagens são usadas para os diferentes *threads* comunicarem entre si. As *queues* de mensagens, pela sua construção, suportam que as mensagens sejam colocadas e retiradas de uma forma assíncrona o que implica que as *threads* mais rápidas não têm de esperar pelas mais lentas. As *queues* funcionam também como uma espécie de área de armazenamento de mensagens que necessitam de processamento.

Para fazer a gestão das *queues*, quando estas estão cheias, escolheu-se o método *Drop-from-head* pois considerou-se que se deveriam penalizar as mensagens mais antigas no sistema em relação às novas. Tanto no SIP como no CAP, caso seja enviada uma mensagem e a resposta a essa mensagem não for recebida, ao fim de um determinado tempo é despoletado um evento. Este evento pode ser um reenvio da mensagem, ou um término da sessão. Desta forma tentou-se minimizar o tempo que uma nova sessão demora a ser processada no IM-SSF quando este se encontra em situação de sobrecarga. O custo a pagar por esta escolha é a perda eventual de algumas sessões mais antigas, o que até pode ser algo positivo pois poderia ser uma destas sessões a causadora da sobrecarga do sistema.

3.2 *Perspectiva Funcional*

Após análise das normas vigentes para o IM-SSF construiu-se um esquema de casos de uso (*Use-Cases)*. Cada caso de uso representa uma funcionalidade utilizável pelos actores. Os actores presentes ilustram as entidades existentes, definidas pela normalização do IMS (*IP Multimedia Subsystem*).

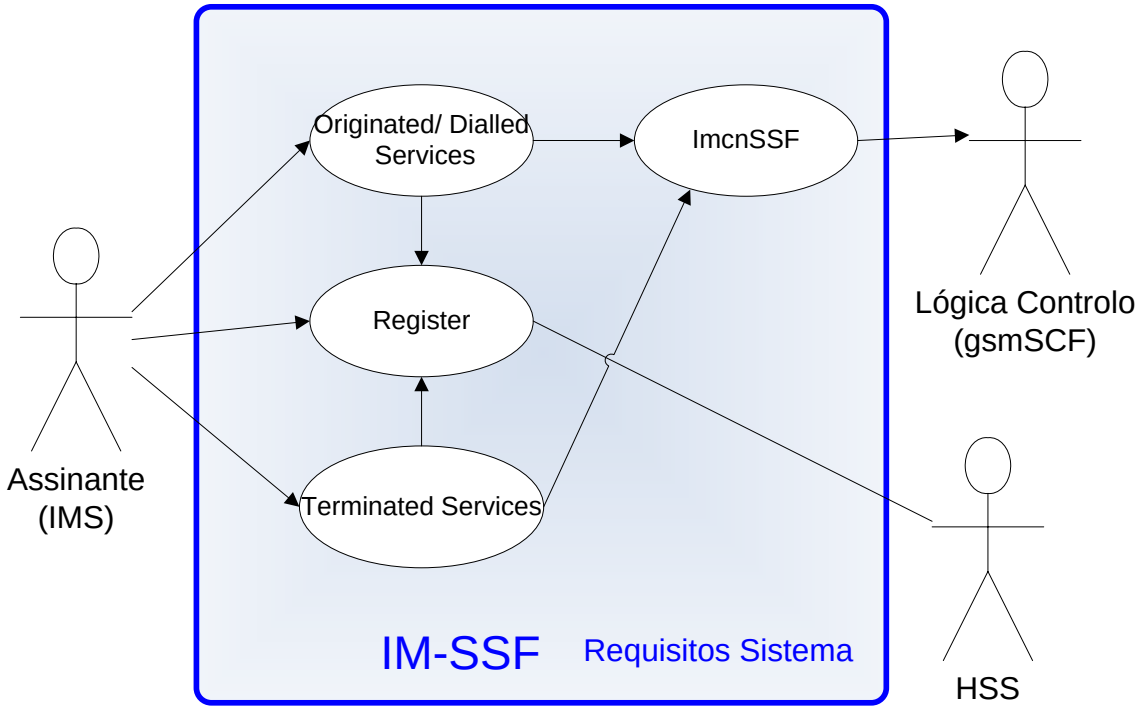

Figura 3-2: Casos de uso do sistema

Na Figura 3-2 é apresentado o diagrama de casos de uso do sistema que pode ser lido da seguinte forma: um assinante pode invocar no IM-SSF um dos seguintes serviços *Originated/Dialled*, *Register*, *Terminated*. O serviço de *Originated/Dialled* pode contactar o serviço de Register (registo do assinante implicitamente durante a chamada). O serviço de *Originated/Dialled* contacta o *gsmSCF* que se encontra fora do sistema IM-SSF sendo assim considerado um actor do IM-SSF. Outro actor é o HSS que será contactado apenas pelo serviço *Register*. O caso de uso *ImcnSSF* representa a gestão da sessão CAP com o *gsmSCF*. As setas apresentadas na Figura 3-2 indicam o sentido de quem inicia a ligação.

Nas telecomunicações são utilizados muito frequentemente, esquemas gráficos para definir inequivocamente o comportamento de sistemas. Uma linguagem que permite este desenho gráfico é a *Specification and Description Language* (SDL), e a norma [3GPP TS 23.278 V6.1.0] fornece vários esquemas detalhados do comportamento do IM-SSF no formato SDL. Estes esquemas estão estruturados de tal forma que são directamente associáveis aos casos de uso identificados para o sistema. Com esta abordagem e programação adequada consegue-se aproximar o programa, dos esquemas SDLs conseguindo relacionar um com o outro. Isto permite que qualquer alteração nos SDLs seja facilmente reflectida no código. Neste caso específico devido ao excelente trabalho efectuado pelo 3GPP nos SDLs e devido à sua fácil leitura, mesmo para pessoas que não conhecem esta linguagem, podem-se utilizar estes esquemas em SDL como uma espécie de mapa do software. A Figura 3-3 apresenta um diagrama SDL simples para o processo de registo.

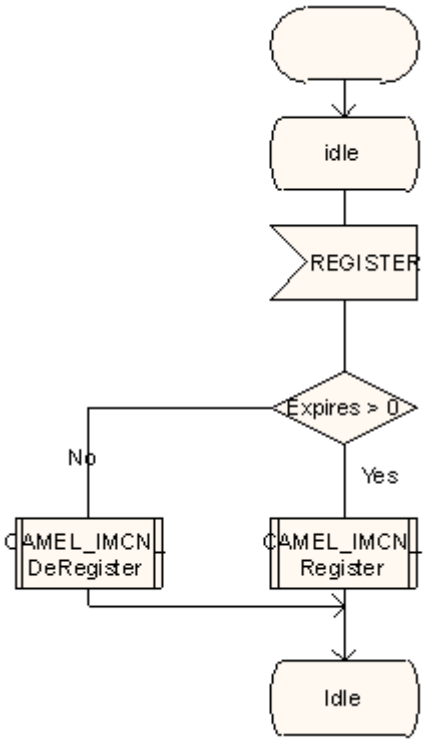

Figura 3-3: SDL do processo de Registo

O diagrama mostra que o sistema começa num estado inicial (*idle*) e com a recepção da mensagem *Register* o sistema vai verificar o campo *expires* da mensagem. Caso este

campo seja maior do que zero, é um processo de registo e por conseguinte a função de *Register* (diagrama SDL que não está aqui exemplificado) é chamada. Caso o campo *expires* seja zero, é invocada a função de *DeRegister*. Depois de executada a função correcta, o sistema volta para o estado inicial ficando novamente pronto para receber novos registos.

3.3 *Perspectiva Lógica*

A perspectiva lógica do sistema IM-SSF é apresentada na seguinte figura.

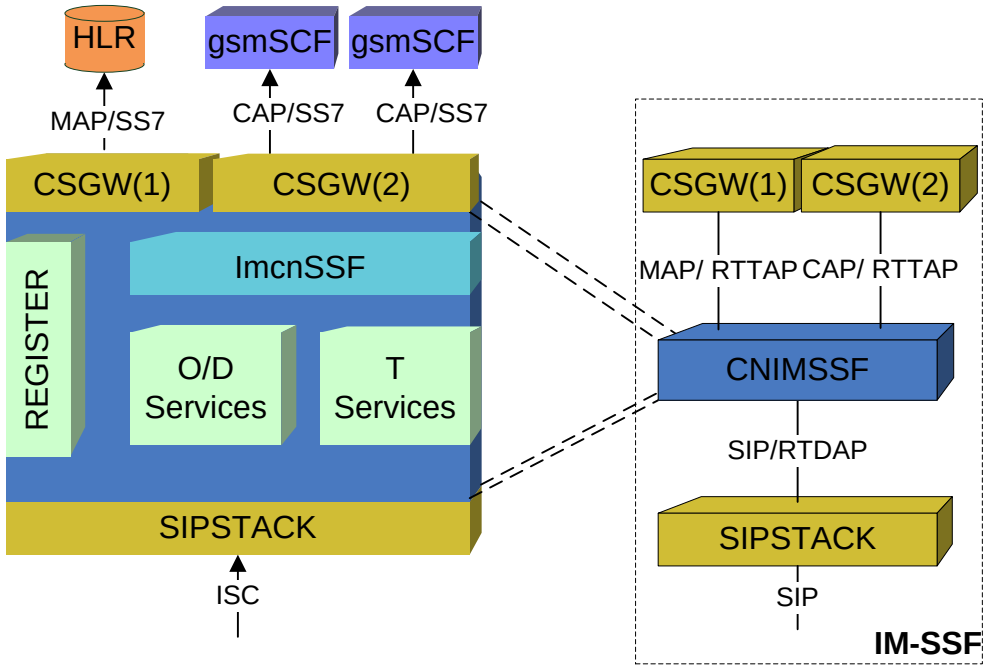

Figura 3-4: Perspectiva lógica do IM-SSF

De seguida serão explicados os vários componentes que juntos compõem a solução do sistema IM-SSF desenvolvido.

3.3.1 SIPSTACK

Este componente é responsável pela comunicação com o S-CSCF através da interface ISC. Baseia-se na utilização de um produto comercial de uma *pilha* SIP (*Session Initiation Protocol*), fornecida pela Trillium [TRILLIUM].

A solução adoptada assenta (i) na pilha SIP Trillium utilizada para descodificar e codificar mensagens SIP numa estrutura e (ii) nas bibliotecas necessárias para a codificação e descodificação de mensagens SIP no protocolo *Real Time Data*

Application Part (RTDAP). Ao CNIMSSF é apenas passada a informação relevante para que este possa identificar os intervenientes na sessão (chamador e chamado) e a sessão propriamente dita. Todos os detalhes de encaminhamento SIP permanecem no módulo SIPSTACK.

Com esta solução é possível receber uma mensagem SIP, descobrir qual a sessão que lhe corresponde e consequentemente codificar a mensagem SIP no protocolo RTDAP para indicar ao CNIMSSF a actualização de estado referente a essa sessão. A solução contempla a situação inversa, ou seja, recebe uma mensagem RTDAP do CNIMSSF e constrói uma mensagem SIP.

3.3.2 O/D Services

Este componente faz parte do módulo CNIMSSF e implementa os fluxos de mensagens definidos na norma [3GPP TS 23.278 V6.1.0] quando é uma sessão que deve ser tratada pelo O/D-IM-BCSM (MO). A decisão de qual o BCSM a utilizar faz parte das competências do S-CSCF (através da distinção MO e MT).

3.3.3 T Services

Este componente faz parte do módulo CNIMSSF e implementa os fluxos de mensagens definidos na norma [3GPP TS 23.278 V6.1.0] quando é uma sessão que deve ser tratada pelo T-IM-BCSM (MT). Este componente é muito semelhante ao O/D-IM-SSF e é aplicado quando o processo de sinalização das sessões se encontra no lado terminado.

3.3.4 REGISTER

Este componente é especializado no tratamento dos registos dos utilizadores no módulo CNIMSSF. Armazena localmente toda a informação referente ao utilizador podendo, caso necessite, recorrer ao HSS ou a configurações internas para obter a informação necessária.

Faz também parte do módulo CNIMSSF e é independente do BCSM a tratar ou seja, este componente está sempre presente e activo quer o CNIMSSF esteja a funcionar em MO ou em MT.

3.3.5 ImcnSSF

Este componente é o elemento principal do CNIMSSF e encontra-se sempre presente, independentemente do BCSM a tratar. Cada instância do *IP Multimédia Core Network Service Switching Function* (ImcnSSF) apenas comunica com um O/D-IM-BCSM ou T-IM-BCSM e nunca com ambos em simultâneo.

Para além da comunicação com esses componentes, é ainda responsável pela implementação de toda a máquina de estados do CAP, sendo também da sua competência a construção das mensagens CAP a enviar, bem como a decisão de qual o serviço a invocar, com base na informação existente no *Register Handling*. Essa comunicação baseia-se em CAP4IMS, que é uma extensão do CAP3 e é transportado em *Real Time Transaction Application Protocol* (RTTAP) para o *CAMEL Signaling GateWay* (CSGW).

3.3.6 CSGW(1)

Este componente é um componente genérico do sistema IM-SSF, responsável pela comunicação SS7 com o HLR /HSS. A sua principal função no IM-SSF é a de disponibilizar essa capacidade de comunicação com o HSS ao *Register Handling*.

3.3.7 CSGW(2)

Este componente é o elemento responsável pela decisão de enviar as mensagens CAP para o *gsmSCF*. Tem a capacidade de comunicar quer em SS7 (interface normalizada definida pelo 3GPP) quer em RTDAP para outros sistemas da PT Inovação.

Os componentes CSGW são na realidade duas instâncias do mesmo componente. Isto deve-se meramente ao objectivo de reduzir a complexidade e aumentar a segurança.

3.4 *CNIMSSF*

Nesta secção é apresentado o componente *Core Network IMSSF* (CNIMSSF) que é o responsável por basicamente implementar a norma [3GPP TS 23.278 V6.1.0] que garante o correcto funcionamento dos serviços identificados nos casos de uso (Figura 3-2: Casos de uso do sistema).

Na Figura 3-5 é apresentado um esquema gráfico que tenta mostrar uma representação visual dos componentes que constituem o módulo CNIMSSF e a forma como estes interagem entre si.

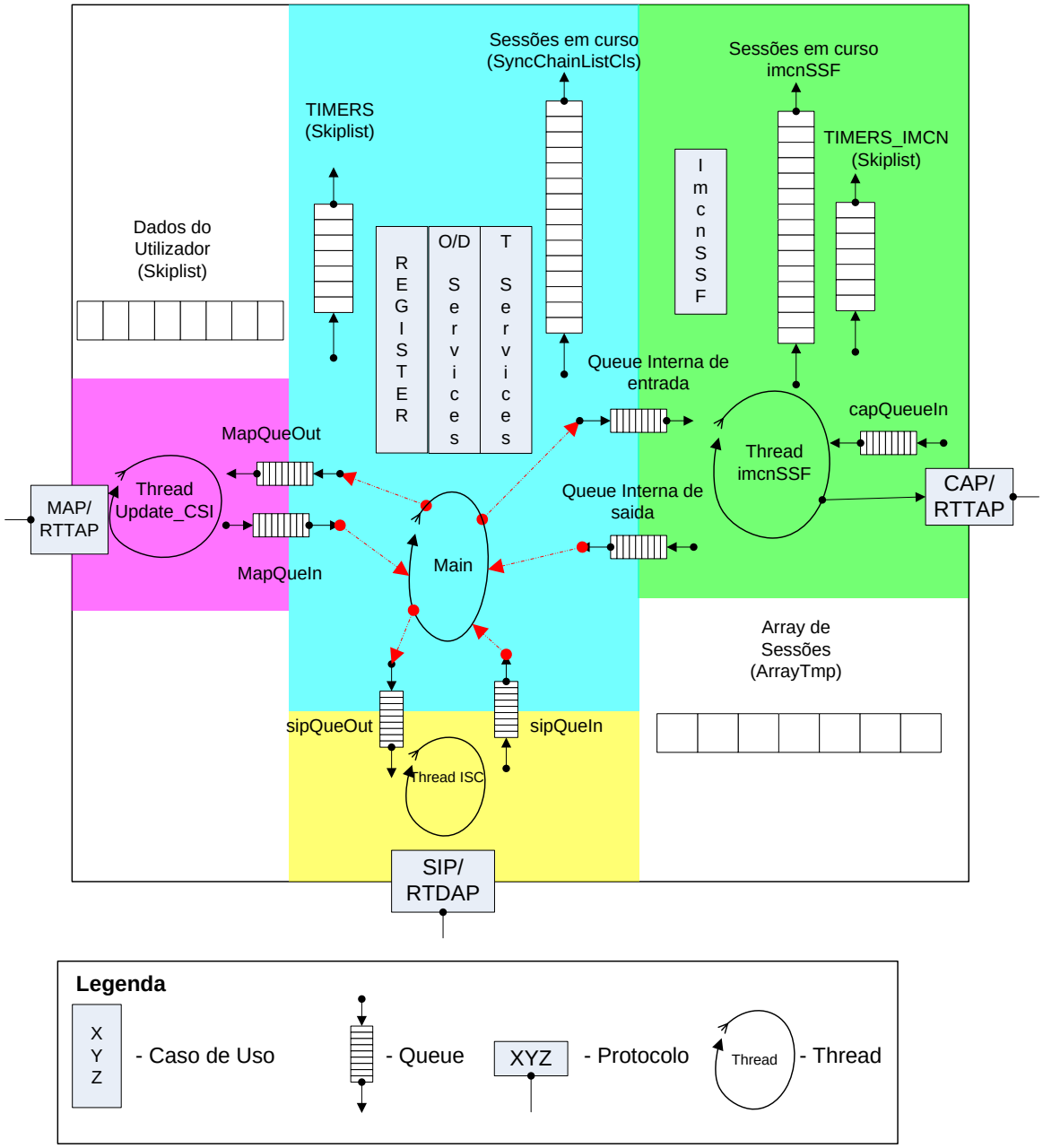

Figura 3-5: Esquema gráfico da solução desenvolvida para o módulo CNIMSSF

A Figura 3-5 apresenta os principais componentes do sistema. Nesta figura consegue-se visualizar a decisão fundamental do desenho do módulo. Esta decisão passa por criar um programa principal (Main) que depois lança uma *thread* para cada interface que necessita de implementar, desacoplando o programa principal das especificidades das

interfaces. Após este desenho decidiu-se o que cada *thread* faria para implementar o correcto funcionamento previsto pela norma [3GPP TS 23.278 V6.1.0].

3.4.1 Interfaces

3.4.1.1 Interface SIPSTACK <-> CNIMSSF

Esta interface é interna ao sistema e possibilita a comunicação do componente CNIMSSF com uma pilha (*stack*) capaz de traduzir mensagens SIP para um tipo de mensagens internas específico e vice-versa. Isto surge devido a uma escolha de arquitectura que pretende ser o mais independente possível. Com esta solução pode-se integrar o componente CNIMSSF com outra pilha SIP (não necessariamente da mesma empresa) mediante algumas pequenas alterações nomeadamente a compreensão desta nova pilha das mensagens enviadas pelo CNIMSSF.

Esta interface é baseada no protocolo RTDAP que é proprietário da PT Inovação e foi desenvolvido para suportar comunicações entre as várias soluções produzidas pela PT Inovação.

O protocolo RTDAP é orientado à mensagem pelo que não é necessário estabelecer uma sessão entre entidades.

As mensagens que circulam por esta interface transportam apenas a informação SIP considerada relevante para a identificação da sessão e seus intervenientes.

3.4.1.2 Interface CNIMSSF <-> CSGW

Nesta interface são utilizadas as mensagens CAP e MAP que são transportadas sobre o protocolo RTTAP. O protocolo RTTAP é uma solução proprietária da PT Inovação e foi especificado como uma simplificação do TCAP para IP. Desta forma é utilizado para transportar mensagens CAP e MAP sobre redes IP, abstraindo-se da máquina de estados imposta no TCAP. Com a utilização deste protocolo, o CSGW consegue fazer a transposição directa das mensagens CAP/RTTAP ou MAP/RTTAP para as redes SS7 onde comunica em CAP/TCAP ou MAP/TCAP respectivamente.

Esta interface é utilizada como MAP/RTTAP quando o CSGW desempenha a função de comunicar com o HLR.

As mensagens CAP/RTTAP circulam quando o CSGW é utilizado para comunicar com o *gsmSCF*.

3.4.2 Elementos fundamentais

Nesta secção são explicados os elementos fundamentais que formam o componente CNIMSSF. Para se compreender melhor os elementos fundamentais é necessário definir dois tipos de listas utilizadas no CNIMSSF: a *SyncChainList* e a *Skiplist*.

A lista do tipo *SyncChainListCls* é uma lista do tipo FIFO onde o acesso é síncrono. Ou seja garante-se que o acesso aos dados só pode ser efectuado apenas por um processo de cada vez. Para efectuar este sincronismo foram utilizados semáforos para sinalizar o acesso aos dados. Outra característica deste tipo de lista é o facto de ser virtualmente infinita pois é baseada numa lista ligada onde se vão criando os elementos À medida que estes vão sendo necessários. Ou seja pode-se iniciar a lista com dez (10) elementos e caso seja necessário colocar um décimo primeiro (11) elemento, a lista cresce para poder colocar esse elemento.

A lista do tipo *Skiplist* é uma lista implementada pela PT Inovação baseada na estrutura de dados descoberta em 1990 por William Pugh. Este tipo de lista é uma estrutura de dados probabilística, baseada em listas ligadas paralelas, com eficiência comparável à de uma árvore binária para a maioria das operações.

Esta lista contém dois elementos: uma chave e o objecto a ser guardado. A *Skiplist* efectua uma espécie de mapeamento entre a chave e o objecto a ser guardado. Quando um par (chave, objecto) é inserido, a chave é comparada com as outras chaves existentes na lista e é inserida na sua posição, garantindo que as chaves na lista se encontram ordenadas. Quando se pretende aceder a um elemento é fornecida a chave e é devolvido o objecto guardado. Na criação deste tipo de listas deve-se fornecer o número máximo de elementos e a chave que indicará o fim da lista. Por exemplo, ao criar uma lista de

dez (10) elementos onde queremos que a chave seja um inteiro não superior a dez (10) a chave a ser colocada como chave final é onze (11).

Dados do Utilizador – Dados do utilizador é uma *Skiplist* que contém a informação CSI (O-IM-CSI, D-IM-CSI e VT-IM-CSI) de cada assinante. Esta informação pode ser acedida através da pesquisa pelo campo *userName* do assinante (ex: NNovo@ptinovacao.pt). Como os dados de informação CSI podem ser extensos, nesta lista é guardado apenas o ponteiro para a localização dos dados. Os dados são armazenados num array que não é descrito na Figura 3-5. Esta solução foi concebida para poder conter a informação necessária dos assinantes, quer esta provenha do HSS ou por leitura de ficheiros de configuração. Esta solução visa possibilitar a existência de perfis de utilizadores com os mesmos CSI partilhados entre eles, diminuindo desta forma os recursos ocupados por cada utilizador registado.

TIMERS – É uma *Skiplist* onde como chave utiliza o tempo actual mais o tempo que pretende aguardar, i.e., quando se pretende adicionar um temporizador de trinta (30) segundos é verificado o tempo actual (*timestamp* devolvido em nano segundos) e somado a este tempo, os trinta (30) segundos respeitando as ordens de grandeza. Com isto é obtida a chave que é inserida na lista. Periodicamente, a *thread* Main retira o primeiro elemento desta lista (o que tem um tempo mais próximo do actual) e compara-o com o tempo actual. Caso o tempo actual seja superior significa que o temporizador inserido na *Skiplist* cuja chave se encontra na primeira posição expirou, logo o Main vai processá-lo.

TIMERS_IMCN – É semelhante ao TIMERS mas esta *Skiplist* é utilizada pela *thread* do *ImcnSSF*.

Array de sessões – É o responsável por guardar toda a informação necessária (informação dos clientes, duração da chamada, etc.) para o bom funcionamento das sessões activas. Cada posição deste array guarda os dados relativos a uma sessão IMS; assim, o número máximo de sessões simultaneamente activas permitidas pelo módulo é dado pelo número máximo de posições deste *array*.

Sessões em curso – É uma lista do tipo *SyncChainListCls* que contém as posições do *array de sessões* que necessitam de ser processadas devido à ocorrência de algum evento relevante à sessão (ex: a chegada de uma mensagem para a sessão em questão). Como quem insere e remove as posições nesta lista é o *Main*, não seria necessário utilizar uma lista deste tipo; contudo, pode-se pretender no futuro colocar outra thread a inserir ou remover posições desta lista (por exemplo, replicar a *thread Main*, para conseguir uma maior utilização de múltiplos processadores)

Na Figura 3-6 pode-se visualizar um exemplo do preenchimento deste tipo de lista.

Existe também uma lista de sessões em curso para a *thread* imcnSSF.

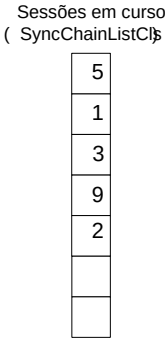

Figura 3-6: Sessões em curso

Na Figura 3-6 pode-se ver que a lista de sessões em curso já se encontra preenchida com alguns valores. O próximo valor a ser retirado da lista é o cinco (5) o que significa que vai processar os dados armazenados na posição cinco (5) do *array* de sessões. Ao inserir um novo valor, por exemplo quatro (4), este será colocado na última posição após o dois (2). Caso se pretendesse colocar o valor três (3) como este já existe seria forçado o processamento dos dados dessa sessão, removendo o três (3) existente entre o um (1) e o nove (9). Após este processamento, o três (3) que se pretendia colocar é inserido no final da lista normalmente.

sipQueOut – É uma *queue* onde se encontra o conteúdo das mensagens SIP/RTDAP para envio. Esta *queue* guarda uma estrutura por cada mensagem que a thread ISC converte para o protocolo RTDAP e transmite para a SIPSTACK, que por sua vez a transforma numa mensagem SIP.

sipQueIn – É uma *queue* onde se encontram as mensagens recebidas em RTDAP da SIPSTACK e que a thread ISC converte convenientemente na estrutura interna do CNIMSSF criada para estas mensagens.

Queue Interna de entrada – É uma *queue* onde se encontram as mensagens internas que são enviadas para a *thread ImcnSSF* processar.

Queue Interna de saída – É uma *queue* onde se encontram as mensagens internas que são enviadas da *thread ImcnSSF* para o *Main*, que irá posteriormente processar de acordo com a máquina de estados respectiva da sessão em causa.

capQueueIn – É uma *queue* onde se encontram as mensagens CAP4IMS recebidas da lógica de controlo CAP através do CSGW(2).

MapQueIn – É uma *queue* onde são colocadas as mensagens enviadas para o HSS, através da interface MAP (*Mobile Application Part*).

MapQueOut – É uma *queue* onde são colocadas as mensagens recebidas do HSS através da interface MAP.

Sessões em curso ImcnSSF – É uma lista de sessões em curso que contém a lista de posições do *array* de sessões que vão ser processadas pela *thread ImcnSSF*. Esta lista é similar à lista sessões em curso usado pela *thread* Main.

3.4.3 Threads

Nesta secção são descritas as *threads* que constituem o CNIMSSF, as suas principais funções e o seu ciclo de processamento.

3.4.3.1 Main

O *Main* é o processo principal do módulo. É este processo que invoca todas as outras *threads* e que inicia todos os processos. É o responsável por processar as chamadas de acordo com a ordem indicada na lista de Sessões em curso. Retira da lista uma posição do *Array de sessões*, vai à respectiva posição no *Array* de sessões e inicia o processo (REGISTER, O/D Services, T Services) que deverá iniciar o processamento.

Entre as competências desta *thread* encontram-se as seguintes:

- Lê os ficheiros de configuração.
- Inicia todos os elementos necessários para o bom funcionamento do módulo CNIMSSF (inclui a criação dos objectos e instanciação das *threads*).
- Verifica e liberta os recursos de uma determinada sessão.

Ciclo de processamento:

A *thread* Main processa X (o X é definido por configuração em NSESSIONPROC) elementos da lista de sessões em curso.

Após o processamento destes X elementos são efectuados os seguintes passos antes do ciclo se repetir novamente:

Passo 1) Retira N (o N é definido por configuração em NUMBER_MSG_GET_INTQUEUOUT) mensagens internas da *queue* interna de saída e coloca-as para processamento (insere a posição do *array* de sessões, indicada na mensagem recebida, na lista de sessões em curso).

Passo 2) Retira M (o M é definido por configuração em NUMBER_MSG_GET_SIPQUEUIN) mensagens SIP da *sipQueIn* e coloca-as para processamento (é colocada a posição do array de sessões, indicada na mensagem retirada da *sipQueIn,* na lista de sessões em curso).

Passo 3) Processa os temporizadores expirados ou seja, retira as entradas da lista de Timers cuja chave seja inferior ao tempo actual e actualiza os dados no array de sessões mediante a máquina de estados utilizada (REGISTER, O/D Services, T Services).

Passo 4) Fica inactiva durante X (o X é definido por configuração em usleep) tempo, para libertar o CPU.

Ao efectuar qualquer um dos passos anteriores, a *thread Main* verifica primeiro se a sessão que se pretende inserir na lista de sessões em curso já lá existe. Caso exista, força o processamento dessa sessão existente na lista. Após este processamento, verifica se o evento que despoletou este comportamento foi a expiração de um temporizador para aquela sessão. Se foi o temporizador que expirou, o facto de se ter processado uma

mensagem que se encontrava à espera de processamento, pode ser o suficiente para parar o temporizador expirado; logo já não é necessário reportar o evento de expiração do temporizador visto este já não fazer sentido no contexto actual.

Caso o processamento forçado não afecte o estado do temporizador, esse temporizador é processado de imediato. Se o evento não foi um temporizador, a sessão à qual corresponde esse evento é colocada no fim da lista de sessões em curso. É de notar que aos eventos proporcionados por temporizadores é dada prioridade máxima, pelo que estes não entram sequer na lista de sessões em curso.

3.4.3.2 Thread ISC

A *thread* ISC é usada para efectuar o processamento das mensagens que chegam pela interface SIP/RTDAP.

Entre as competências desta *thread* encontram-se as seguintes:

- Detecta se a mensagem recebida pela interface SIP/RTDAP implica a criação de uma nova sessão e em caso afirmativo, atribui uma posição do *array* de sessões para esta nova sessão.
- Guarda as mensagens SIP/RTDAP recebidas e descodificadas na *queue sipQueIn.*
- Relaciona o parâmetro SIP *Call-ID* da mensagem e a posição reservada no array de sessões. O parâmetro SIP *Call-ID* identifica a sessão SIP inequivocamente; ao associar uma posição do array de sessões a este parâmetro, consegue-se identificar a posição do array de sessões para todas as mensagens aqui recebidas (SIP/RTDAP).
- Indica a qual processo se refere a mensagem SIP recebida (REGISTER, O/D Services, T Services). Isto efectua-se consultando um parâmetro de configuração que indica qual o modo de funcionamento do IM-SSF, MO ou MT. Caso a mensagem recebida seja um RTDAP *Register* equivalente ao *SIP* "Register" então o processo a ser invocado é o REGISTER.

- Retira mensagens da *queue sipQueOut*, codifica-as no protocolo RTDAP e envia-as para o módulo SIPSTACK.

Ciclo de processamento:

Esta *thread* executa em ciclo infinito o seguinte:

- Efectua X (o X é definido por configuração em *MAX_SIP_POOLS*) tentativas consecutivas para receber dados do s*ocket,* convertendo-os em mensagens (estruturas internas) e colocando-as na *queue* de mensagens *sipQueIn*, executando para cada mensagem os seguintes passos:

  Passo 1) Verifica a qual posição do *array* de sessões a mensagem pertence. Caso seja uma nova sessão é reservada uma nova posição no array de sessões para essa sessão e são iniciados todos os parâmetros existentes na posição encontrada.

  Passo 2) Indica a qual processo se refere a mensagem SIP recebida (REGISTER, O/D Services, T Services).

  Passo 3) Fica inactiva durante X (o X é definido por configuração em usleep) tempo, para libertar o CPU.

- Tenta retirar consecutivamente X (o X é definido por configuração em MAX_SIP_SENT) mensagens da *queue* de mensagens *sipQueOut*, codifica-as no protocolo RTDAP e envia-as para o destino.
- Verifica se a ligação com o módulo SIPSTACK foi terminada e em caso afirmativo, activa o procedimento de segurança Terminação de Sessão involuntária.

3.4.3.3 Thread ImcnSSF

A *thread* ImcnSSF é responsável por manter o contexto das sessões CAP associadas às sessões SIP. Também tem como responsabilidades enviar/receber mensagens CAP/RTTAP. No leque de funções desta *thread* encontra-se o seguinte:

- Comunica usando CAP;

- Prepara os *Detection Point* (DP) de acordo com as instruções recebidas (ao receber um *"Request Report BCSM Event"*, via interface CAP);
- Fornece instruções à *thread* responsável pelas sessões SIP (*Main)* através de mensagens internas que são enviadas pela *queue* interna de saída;
- Insere e processa elementos da lista de sessões em curso imcnSSF;
- Processa as mensagens provenientes da *queue* interna de entrada e da *queue capQueueIn*;

Ciclo de processamento:

A thread ImcnSSF processa X elementos (o X é definido por configuração em *NCALLPROCIMCN*) da lista de sessões em curso imcnSSF. Após o processamento destes X elementos são efectuados os seguintes passos antes de o ciclo se repetir novamente:

Passo 1) Processa os temporizadores CAP expirados (da lista TIMERS_IMCN).

Passo 2) Tenta receber N (o N é definido por configuração em NUMBER_MSG_GET_CAPQUEUEIN) vezes consecutivas, dados do socket CAP, transformando estes dados em mensagens e colocando-as para processamento (insere a posição da chamada correspondente na lista de sessões em curso imcnSSF).

Passo 4) Retira M (o M é definido por configuração em NUMBER_MSG_GET_INTQUEUEIN) mensagens da *queue* interna de entrada e coloca-as para processamento.

Passo 5) Fica inactiva durante X (o X é definido por configuração em usleep) tempo, para libertar o CPU.

A inserção das sessões na lista de sessões em curso é efectuada de uma forma semelhante à explicada para a *thread* Main. Desta forma, garante-se que na lista de sessões cada entrada é única, ou seja, não é possível ter duas ou mais vezes repetida a mesma sessão na lista. Quando se coloca uma sessão para processamento, é colocado no

array de sessões uma indicação de qual o tipo de evento que despoletou o processamento (temporizadores, mensagem CAP/RTTAP, mensagem SIP/RTDAP, mensagem MAP/RTTAP). De acordo com esta informação, quem processa a sessão em causa pode localizar os dados que foram actualizados pelo evento que ocorreu. Assim, por exemplo, se chegarem duas mensagens CAP consecutivas para a mesma sessão a primeira actualiza os dados no *array* de sessões e fica à espera de processamento. Quando chega a segunda mensagem CAP, é forçado o processamento da primeira mensagem, após o qual são actualizados os dados do *array* de sessões e insere-se a posição do *array* de sessões na lista de sessões em curso imcnSSF para que os dados sejam processados.

Temporizadores:

A *thread* ImcnSSF possui um conjunto de temporizadores que usa para controlar as sessões CAP. Os temporizadores são os seguintes:

Tssf — Sempre que é enviado um pedido, é activado este temporizador; caso ele expire, é porque algo se passou com o *gsmSCF* e deve-se efectuar algum procedimento relativo à sessão que está em curso.

Tcp — Este temporizador conta a duração de um período de uma sessão.

Tsw — Temporizador que quando expira indica que deve ocorrer uma mudança de tarifa. Este temporizador é activado pela mensagem "Apply Charging".

Tccd — Este temporizador é iniciado quando é enviado um "Apply Charging Report" e é parado quando é recebido um "Apply Charging". Desta forma a expiração deste temporizador indica que ocorreu algum problema relacionado com a taxação no *gsmSCF,* logo deve ser tomada alguma medida.

Tw — Quando este temporizador expira, é tocado um toque de aviso ao cliente. Este temporizador é utilizado por exemplo, para indicar ao cliente que o final de saldo está para ocorrer, através do toque de um sinal sonoro.

Adicionalmente, a *thread* ImcnSSF possui a variável de tempo DELTA. Esta variável é usada para guardar o instante de envio do "Apply Charging Report" e permite contabilizar a diferença de tempo entre o seu envio e a recepção do "Apply Charging" seguinte.

3.4.3.4 Thread Update_CSI

A *thread* Update_CSI é responsável pela comunicação com o HSS através da(s) interface(s) SI/SH usando MAP/Diameter, respectivamente.

Entre as competências desta *thread* encontram-se as seguintes:

- Actualiza os dados do cliente na lista de dados do utilizador de acordo com as mensagens recebidas.
- Codifica e descodifica as mensagens recebidas na queue *MapQueOut* no protocolo MAP/RTTAP.
- Gere a ligação MAP/RTTAP.

Ciclo de processamento:

Esta *thread* executa em ciclo infinito o seguinte:

- Tenta retirar consecutivamente X (X é definido por configuração em MAXNUMBEROFPROCESSCAP) dados do *socket* convertendo-os em mensagens e efectuado o seguinte passo por cada mensagem descodificada:

  Passo 1) Descodifica os dados numa estrutura inteligível pelo IM-SSF.

  a. Se for uma mensagem "Notify Subscriber Data Change" actualiza os dados do cliente e responde com um "Notify Subscriber Data Change Ack"

  b. Se for uma outra mensagem qualquer

     i. Coloca a mensagem recebida, já devidamente descodificada, na fila de mensagens *MapQueIn*

- Retira mensagens da queue *MapQueOut,* codifica-as e envia-as para o HSS.

- Fica inactiva durante X (o X é definido por configuração em usleep) tempo, para libertar o CPU.

3.4.4 Máquina de estados

De seguida são descritos os processos responsáveis pelo funcionamento do sistema. Estes processos conseguem-se relacionar directamente com os casos de usos definidos para o sistema.

REGISTER – processo responsável pelo registo e fim do registo do assinante no módulo CNIMSSF.

Contacta a base de dados HSS através da interface SI/SH de forma a obter a informação CSI para o respectivo assinante e aguarda essa informação na lista de dados do utilizador.

O/D Services – Responsável pelo processamento de chamadas originadas por um assinante que pertence à rede IMS no qual o sistema IM-SSF se encontra. É nesta máquina de estados que são aplicados os CSIs O_IM_CSI e D_IM_CSI, Esta máquina de estados é invocada quando o sistema IM-SSF se encontra a funcionar no modo MO.

T Services – Responsável pelo processamento de chamadas com destino a um assinante que pertence à rede IMS no qual o sistema IM-SSF se encontra. Esta máquina aplica o CSI VT_IM_CSI.

3.4.5 Mecanismos de prevenção

3.4.5.1 Terminação de Sessão involuntária

Quando surge algum problema com o componente CNIMSSF ou com uma das suas sessões, é utilizada uma lista (*disconect List*) semelhante à lista de sessões em curso, onde se marcam as sessões que devem ser libertadas. Esta lista só é utilizada em casos de problemas, tais como:

- Uma *queue* encontra-se cheia e é necessário remover uma mensagem. A sessão relativa a essa mensagem é adicionada a esta lista, com a finalidade de terminar a sessão em causa.

- A ligação com a SIPSTACK é perdida. Neste caso, é feita uma imagem das sessões activas no momento e essas sessões são colocadas nesta lista para serem terminadas.
- Ao tentar enviar uma mensagem CAP, se ao fim de algumas tentativas consecutivas não se consegue efectuar o envio. Neste caso, a mensagem é perdida e a sessão desta mensagem é adicionada a esta lista para ser terminada.

3.4.5.2 Congestionamento de *queues*

Quando as *queues* usadas no sistema atingem o seu limite, irá necessariamente ocorrer perda de mensagens. Nesta situação, foi decidido implementar o sistema *Drop-from-head* onde a mensagem a ser descartada é a do início da fila de espera, como já foi indicado em 3.1.2.2 . Isto significa que as mensagens mais antigas são descartadas o que provoca que as chamadas que já estão pendentes no sistema há mais tempo sejam perdidas (terminadas com insucesso). Pretende-se com este método privilegiar as novas sessões reduzindo as situações de indisponibilidade do serviço.

Capítulo 4 - Testes

A fase de testes é de extrema importância pois é com esta fase que se testa o correcto funcionamento do módulo segundo as normas que se lhe aplicam e conforme as estratégias pensadas na sua concepção e aplicadas no desenvolvimento. Neste capítulo, são descritos dois tipos de testes: testes comprovativos do correcto funcionamento do sistema e testes de desempenho que aferem a eficácia das soluções e estratégias adoptadas. Decidiu-se efectuar a separação entre os testes funcionais e os testes de desempenho pois, embora os cenários sejam idênticos, os objectivos e parâmetros analisados são diferentes e, por conseguinte, conduzem a conclusões diferentes.

4.1 *Performance e Alta Disponibilidade*

Em qualquer sistema real podem acontecer falhas e os sistemas devem ser concebidos de forma a minimizar o seu impacto. No sistema desenvolvido no âmbito desta dissertação, é dada prioridade à disponibilidade do sistema em detrimento dos seus recursos tais como a memória por exemplo. Com esta estratégia pretende-se que qualquer falha num componente do sistema seja rapidamente corrigida e solucionada de forma que os novos pedidos sejam continuamente atendidos, minimizando o impacto dessa falha nos clientes. Na Figura 4-1 é apresentada uma solução de alta disponibilidade distribuída por duas máquinas que estão diferenciadas pelo esquema de cores adoptado (azul – máquina 1, amarelo – máquina 2).

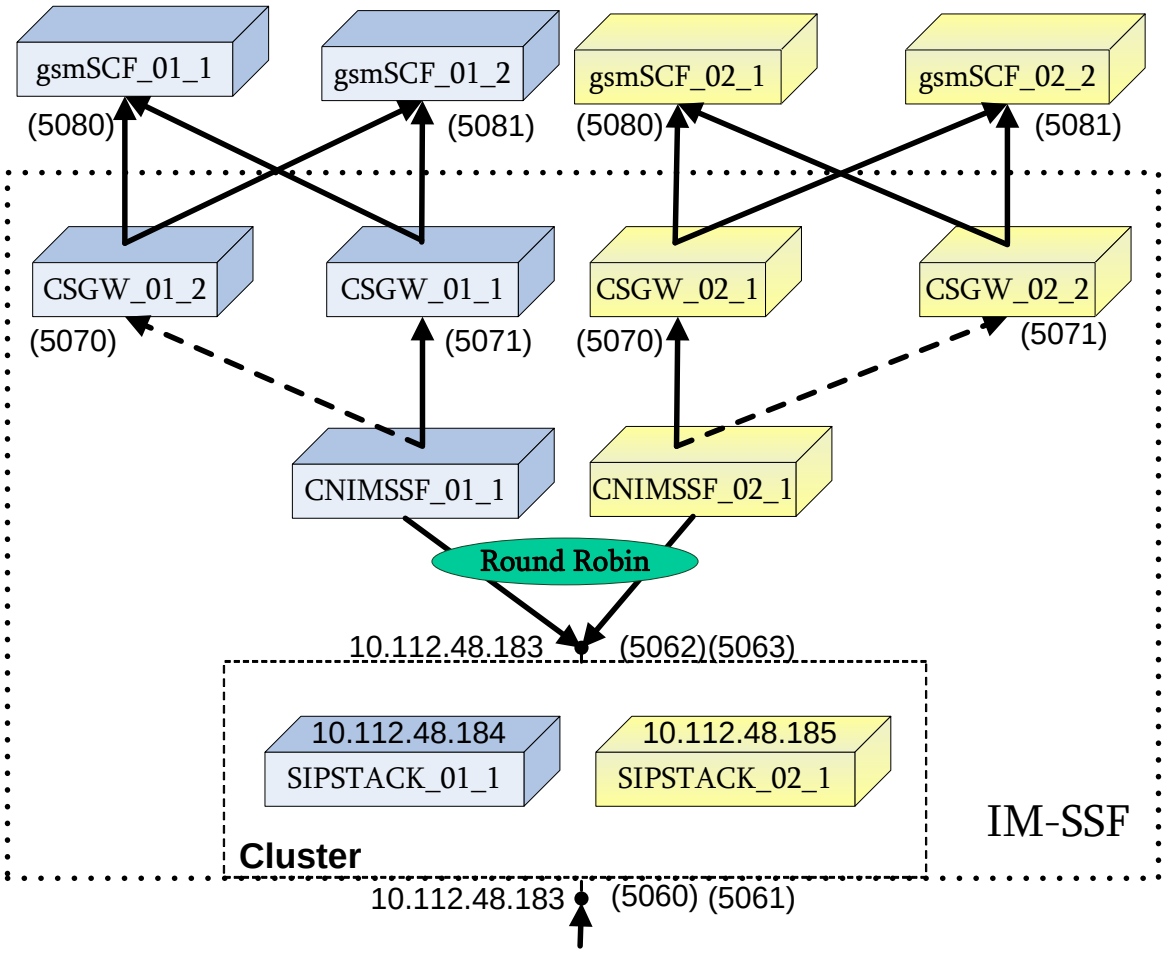

Figura 4-1: Arquitectura de Alta Disponibilidade

Para garantir disponibilidade de serviço em caso de falhas, desenharam-se vários mecanismos de segurança. Para o módulo SIPSTACK (que troca mensagens SIP com o S-CSCF da rede IMS e troca mensagens SIP/RTDAP com o CNIMSSF) é necessário que o seu endereço IP e respectivos números de porto se mantenham ao longo do tempo para garantir a disponibilidade do serviço. Para garantir este requisito, utilizou-se um sistema de *clustering* que permite atribuir um endereço IP virtual a uma máquina e no caso de falha desta, comutar para outra máquina mantendo o mesmo IP virtual. A Figura 4-1 especifica os endereços IP reais das duas máquinas pertencentes ao *cluster* (10.112.48.184 e 10.112.48.185) e o endereço IP virtual (10.112.48.183) usado pelo *cluster* sendo este o conhecido pelo S-CSCF e pelo CNIMSSF.

De acordo com o assumido no desenvolvimento do sistema IM-SSF este possui duas instâncias para proporcionar os seus dois modos de funcionamento MO e MT (secção

3.1). Desta forma a arquitectura de alta disponibilidade é replicada para cada modo de funcionamento. Para dividir os dois cenários MO e MT decidiu-se usar portos distintos para cada cenário, o porto 5060 utilizado pelo S-CSCF invoca o cenário MO e o porto 5061 o cenário MT. Por outro lado o porto 5062 é utilizado pelo CNIMSSF para estabelecer comunicação com a SIPSTACK. Na Figura 4-1 apresentam os módulos utilizados para construir apenas o cenário MO por uma questão de simplicidade não estão presentes os módulos MT mas no sistema implementado todos os módulos são replicados e correm noutros portos (no cluster a SIPSTACK para MT corre nos portos 5061 e 5063 para contacto do S-CSCF e CNIMSSF).

O módulo CNIMSSF é o ponto de maior inteligência do sistema e em caso de falha deste, todas as sessões são terminadas e o serviço fica indisponível. Assim, optou-se por tornar a SIPSTACK num servidor para o CNIMSSF e desta forma podem-se ter vários módulos CNIMSSFs ligados a um único módulo SIPSTACK onde as sessões são distribuídas segundo uma filosofia de *Round-Robin* entre cada um deles (na Figura 4-1, o sentido de uma ligação indica a estratégia cliente-servidor adoptada: a origem é o cliente e o destino é o servidor). Na situação de falha de um dos módulos CNIMSSF, o módulo SIPSTACK liberta todas as sessões servidas por esse CNIMSSF.

Quanto ao CSGW, numa situação de falha, e uma vez que este módulo não guarda qualquer informação relevante, optou-se por uma solução de *Active/Standby*, em que um segundo CSGW assume o modo activo, quando o primeiro CSGW falha. Esta comutação é proporcionada pelo CNIMSSF que conhece à partida os dois CSGW's e apenas utiliza um deles para efectuar a comunicação CAP. O CSGW consegue ligar-se a um ou mais *gsmSCF*. Para distinguir qual dos *gsmSCF* deve contactar o CSGW possui uma lista configurável que, de acordo com a identificação do serviço, encaminha os pedidos para o *gsmSCF* correspondente. O *gsmSCF* não faz parte da solução do IM-SSF e é representado apenas para enfatizar o facto de o CSGW poder contactar mais do que um *gsmSCF* de acordo com o serviço invocado. Para testes, usou-se um módulo pertencente à PT Inovação que é capaz de enviar os fluxos de mensagens CAP tal como

faria um *gsmSCF*. Na arquitectura de alta disponibilidade apresentada, os *gsmSCF* pertencem à mesma máquina onde corre o CSGW que o contacta. Isto deve-se ao facto desta arquitectura ter sido utilizada para efectuar os testes de desempenho para um cliente do IM-SSF que propôs este cenário de acoplamento do *gsmSCF* ao sistema IM-SSF desenvolvido. De acordo com isto, os valores obtidos em termos de tempos poderiam ser ligeiramente maiores se as mensagens tivessem de passar efectivamente pela rede.

Todos os testes que são apresentados ao longo deste capítulo foram efectuados em máquinas com as seguintes especificações:

IP	*Nome*	*Processador*	*Memória*	*Sistema Operativo*
10.112.48.115	Twin2	4x Intel Xeon 2.4GHz	5GigaBytes	REL 3
10.112.48.184	Dscp1	4x Intel Xeon 3.4GHz	2GigaBytes	REL 4
10.112.48.185	Dscp2	4x Intel Xeon 3.4GHz	2GigaBytes	REL 4

Tabela 4-1: Características das máquinas usadas

Na Tabela 4-1 REL *x* significa ***Red Hat Enterprise Linux*** versão *x* e 4x *Intel Xeon y.z* GHz significa que a máquina possui quatro processadores *Intel Xeon* que funcionam à frequência de *y.z* GHz cada.

Dscp1 e Dscp2 são as máquinas onde se encontra instalado o sistema IM-SSF segundo a arquitectura de alta disponibilidade apresentada na Figura 4-1. A Twin2 corre os utilizadores, chamador e chamado, utilizados para efectuar os testes.

Para os testes funcionais utilizou-se a seguinte configuração de máquinas (grande parte desta informação já foi previamente especificada na Figura 4-1):

Módulo	*IP*	*Porto*
gsmSCF_01_1	10.112.48.184	5080
gsmSCF_01_2		5081
gsmSCF_02_1	10.112.48.185	5080
gsmSCF_02_2		5081
CSGW_01_1	10.112.48.184	5070
CSGW_01_2		5071
CSGW_02_1	10.112.48.185	5070
CSGW_02_2		5071
CNIMSSF_01_1	10.112.48.184	5062
CNIMSSF_02_2	10.112.48.185	5062
SIPSTACK_01_1	10.112.48.184	5060
Chamador (1)	10.112.48.115	5060
Chamado (1)		5061
Chamador (2)	10.112.48.115	5062
Chamado (2)		5063
Chamador (3)	10.112.48.115	5064
Chamado (3)		5065

Tabela 4-2: Ambiente de teste

4.2 *Testes funcionais*

A fase de testes é uma fase importante na concepção de qualquer módulo. Esta é uma fase iterativa nos primeiros tempos enquanto o módulo não atinge uma fase estável, onde cumpra os requisitos propostos.

Os testes funcionais têm como objectivo validar o cumprimento dos requisitos especificados para o módulo.

Como as operadoras onde se pretende vender este sistema não fornecem uma interface MAP com o HSS, a *thread Update CSI* perdeu o seu valor, sendo desactivada. Assim os perfis dos utilizadores registados são carregados com base em ficheiros de configuração.

Com base nos diagramas SDL da norma [3GPP TS 23.278 V6.1.0] extrapolaram-se alguns testes para o sistema IM-SSF. Outros testes, como o teste 1.1 HA, são testes que a

PT Inovação impõem aos seus sistemas. A Tabela 4-3 apresenta os testes do sistema IM-SSF.

Grupo	*Nome*	*Propósito*
1	GR	Validar os requisitos globais do IM-SSF
1.1	HA	Validar a solução de Alta Disponibilidade
2	REG	Registo e fim de registo no IM-SSF
2.1	REG_SUB_REG	Registo no IM-SSF
2.2	REG_SUB_UNREG	Fim de registo no IM-SSF
2.3	REG_AUTO_REG	Registo automático no IM-SSF (MO)
2.4	REG_IMP_REG	Registo implícito no IM-SSF (MT)
3	O	Controlo de sessões originadas
3.1	O_IM_CSI_STATIC	Validação do O-IM-CSI estático
3.2	O_IM_CSI_UNCOND	Validação do critério incondicional (*unconditional*) no O_IM_CSI
3.3	O_IM_CSI_EN_LIST	Validação da lista de critérios de permissão do O_IM_CSI
3.4	O_IM_CSI_IN_LIST	Validação da lista de critérios de inibição do O_IM_CSI
3.5	O_IM_CSI_EN_IN_LIST	Validação da lista de critérios de permissão e inibição do O_IM_CSI
3.6	O_IM_CSI_DCH	Validação do tratamento de chamadas por defeito no O_IM_CSI
3.7	O_BCH	Validação do tratamento básico no lado originado
3.8	O_CCD	Validação do controlo de duração das sessões no lado originado
4	D	Controlo de sessões onde o destino tem um perfil bem conhecido
4.1	D_IM_CSI_STATIC	Validação do D_IM_CSI estático
4.2	D_IM_CSI_DIALL	Validação do critério de análise dos números marcados do D_IM_CSI
4.3	D_IM_CSI_DCH	Validação do controlo da sessão por defeito no D_IM_CSI
4.4	D_BCH	Validação do controlo de sessões básico no *Dialled*
5	T	Controlo de sessões terminadas
5.1	VT_IM_CSI_STATIC	Validação do VT_IM_CSI estático
5.2	VT_IM_CSI_UNCOND	Validação do critério incondicional do VT_IM_CSI

Grupo	*Nome*	*Propósito*
5.3	VT_IM_CSI_DCH	Validação do controlo de sessões por defeito do VT_IM_CSI
5.4	T_BCH	Validação do controlo de sessões básico no lado terminado
5.5	T_CCD	Validação do controlo da duração de sessões no lado terminado

Tabela 4-3: Lista de testes de módulo

Os testes identificados por 1,2,3,4 e 5 são grupos que identificam um conjunto de testes. De seguida, descreve-se o que é cada um dos testes e qual o resultado da sua realização no sistema desenvolvido.

Grupo 1:

1.1– Este teste permite validar a solução de alta disponibilidade. Para isso foram executados os seguintes passos

Passo 1 – o CSGW activo foi abortado e verificou-se que o CNIMSSF se ligou ao segundo.

Passo 2 – o CNIMSSF foi abortado e verificou-se que a SIPSTACK reencaminhou todo o tráfego para o outro CNIMSSF que ainda se encontrava activo. Nesta situação a SIPSTACK terminou todas as sessões que mantinha para o CNIMSSF que foi abortado.

Passo3 – a SIPSTACK foi abortada e verificou-se que o *cluster* arrancou uma nova SIPSTACK. Nesta situação também se verificou que os CNIMSSFs começaram a libertar as sessões que tinham activas.

Grupo 2:

2.1 – Neste teste foi enviado um SIP "REGISTER" para o sistema e através da análise das mensagens impressas pelo CNIMSSF verificou-se que o registo foi efectuado.

2.2 - Neste teste enviou-se um SIP "REGISTER" onde no campo SIP *Expires* se encontrava o valor 0 (zero). Isto significa que é um fim de registo. Analisando

as mensagens impressas pelo CNIMSSF verificou-se que este termina o registo do cliente.

2.3 – Neste ponto verificou-se a opção que se colocou no CNIMSSF que permite que um utilizador não registado seja registado automaticamente no CNIMSSF quando tenta efectuar uma chamada.

2.4 – O mesmo que no teste 2.3 mas para quando o CNIMSSF se encontra a operar no modo MT. Esta situação ao contrário do teste 2.3 encontra-se na norma, pois para serviços terminados o IM-SSF pode não receber o pedido de registo SIP ("REGISTER"*).*

Grupo 3:

3.1 – Neste teste verificou-se que o ficheiro onde se encontram os parâmetros O_IM_CSI é lido correctamente. Este é um teste que serve para confirmar o correcto funcionamento da solução encontrada para obter os CSI dos clientes quando o sistema IM-SSF não possui forma de contactar o HSS.

3.2 – Este teste verificou que os critérios que permitem ou não o serviço O_IM_CSI são lidos e usados correctamente. Neste teste foram verificados os DP estáticos configurados no ficheiro O_IM_CSI. Os DP estáticos existentes no O_IM_CSI são: *Collected_Info* e *Route_Select_Failure*.

3.3 – Neste teste validou-se a lista que verifica se o destino chamado pelo utilizador pode accionar algum serviço (*enabling*). Quando o destino está incluído nesta lista, o IM-SSF contacta o gsmSCF para controlar a sessão. Esta lista serve por exemplo para indicar se em determinada sessão é necessário o controlo proporcionado pelo gsmSCF.

3.4 – Neste teste validou-se a lista que permite verificar se a sessão não deve contactar o gsmSCF por análise do destino (*inhibiting*). Quando o destino é incluído nesta lista e não é incluído na lista anterior (3.3), o IM-SSF não contacta o gsmSCF. O processo de validação das listas é o seguinte: quando um número se encontra na lista validada em 3.3 é contactado o gsmSCF, caso

contrário é verificada a lista de inibir. Se o número se encontrar na lista de inibir o gsmSCF não é contactado, caso contrário, é contactado o gsmSCF.

3.5 – Este teste permitiu verificar que quando o utilizador tenta efectuar uma sessão, não se encontrando nem na lista de permissão (*enabling*) nem na lista de inibição (*inhibiting*) o gsmSCF é contactado. Com estas listas é possível validar se determinadas sessões devem ou não contactar o gsmSCF.

3.6 – Este teste efectuou a validação do parâmetro *Default Call Handling* que é consultado quando o IM-SSF não consegue contactar com o *gsmSCF*. Este parâmetro pode ser configurado para deixar continuar a sessão ou para a terminar.

3.7 – Este teste permitiu verificar que o IM-SSF, quando está a operar em MO consegue, por ordem do *gsmSCF,* terminar sessões, mudar o destino, continuar a sessão, armar pontos de detecção. Este teste serve para verificar se o IM-SSF segue correctamente as ordens do *gsmSCF* tal como a especificação ([3GPP TS 23.278 V6.1.0]) o indica.

3.8 – Este teste permitiu validar que o IM-SSF quando em funcionamento MO, efectua a taxação correctamente. Neste teste, é estabelecida e terminada uma sessão e é verificado se o tempo reportado pelo IM-SSF como tempo de duração de uma sessão é o correcto.

Grupo 4:

4.1 - Da mesma forma que se validou o O_IM_CSI lido de um ficheiro este teste validou que o ficheiro D_IM_CSI é lido correctamente.

4.2 – O D_IM_CSI contém uma lista de critérios que são comparados com o número digitado para a identificação do serviço a aplicar. Neste teste foi validado que os critérios são aplicados correctamente. Para esta validação foi efectuada uma sessão onde o destino seria um número verde. Após a análise destes critérios foi contactado um *gsmSCF* para fornecer o serviço

propriamente dito (mudança do destino para um outro número). O D_IM_CSI contém apenas um DP estático o *Analysed_Information.*

4.3 – Este teste permitiu validar o parâmetro *Default Call Handling* existente no ficheiro D_IM_CSI, da mesma forma como foi feito no teste 3.6 efectuado para o O_IM_CSI.

4.4 – Este teste verificou que numa sessão de controlo *Dialled* aberta com o *gsmSCF*, o IM-SSF consegue receber e processar correctamente as instruções do *gsmSCF* de acordo com a especificação [3GPP TS 23.278 V6.1.0].

Grupo 5:

5.1 – Neste teste foi verificada a correcta leitura do ficheiro VT_IM_CSI.

5.2 – Este teste permitiu verificar que os DP estáticos configurados no ficheiro de configuração são aplicados correctamente. Os DP estáticos associados ao VT_IM_CSI são: *Termination_Attemp_Authorised*, *T_Busy* e *T_No_Answer*.

5.3 – Este teste permitiu validar o parâmetro *Default Call Handling* existente no ficheiro VT_IM_CSI, da mesma forma como foi feito no teste 3.6 efectuado para o O_IM_CSI.

5.4 – Este teste verificou que numa sessão de controlo para o terminado, estabelecida com o *gsmSCF,* o IM-SSF consegue receber e processar correctamente as instruções do *gsmSCF* de acordo com a especificação [3GPP TS 23.278 V6.1.0]

5.5 – Neste teste foi verificado que a informação de taxação produzida pelo IM-SSF com destino ao *gsmSCF* é correcta quando o IM-SSF funciona no modo MT. Neste teste é efectuada uma sessão e após o estabelecimento da sessão é verificado que enquanto a sessão permanece activa o IM-SSF vai enviando mensagens de *Apply Charging Report* como indica a especificação.

Na Figura 4-2 é apresentada a arquitectura de testes onde se diferencia o IM-SSF quando este está a funcionar em MO ou em MT. O IM-SSF em funcionamento MO é

invocado pelo S-CSCF que serve o originador da sessão. O IM-SSF no funcionamento MT é invocado pelo S-CSCF que serve o destino da sessão.

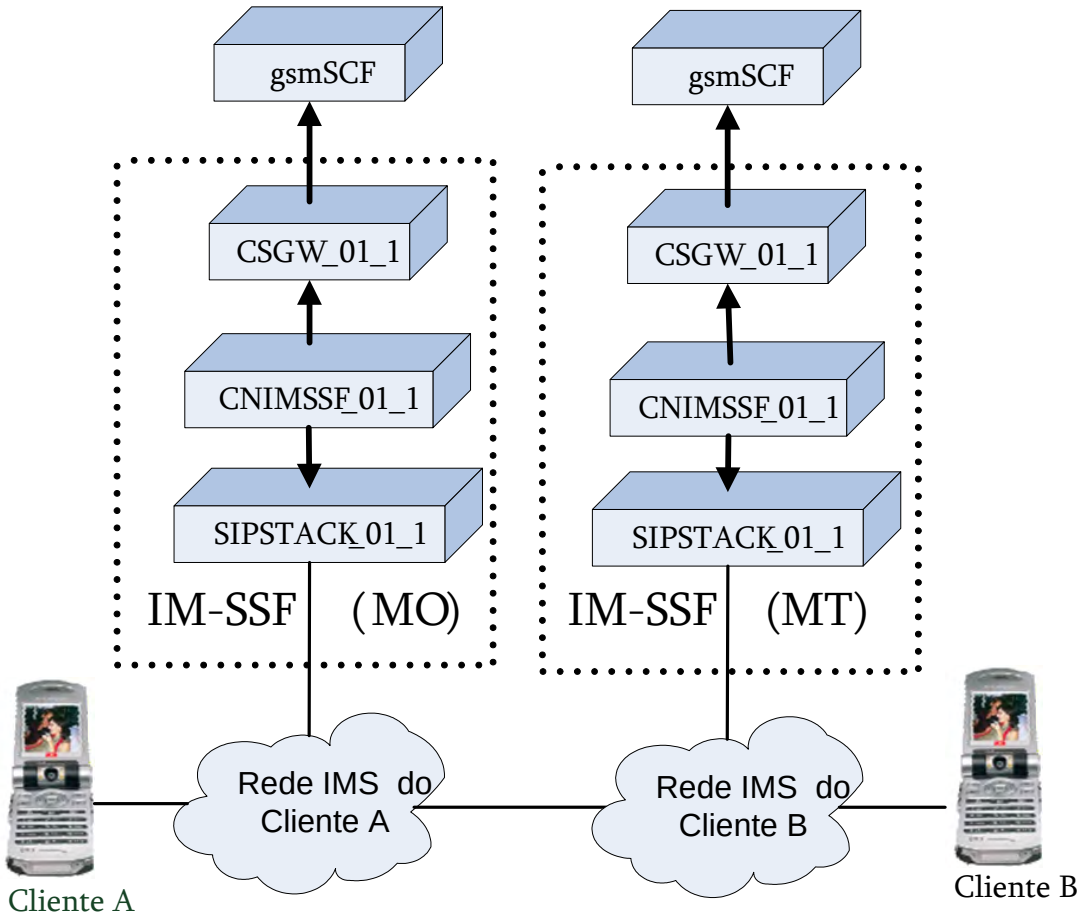

Figura 4-2: Arquitectura de testes

Nas chamadas MO, o sistema na pior das hipóteses activa o serviço de O_IM_CSI e o de D_IM_CSI antes de prosseguir com a sessão (a melhor hipótese seria invocar apenas um dos dois serviços). Estes dois serviços activos implicam duas invocações ao *gsmSCF* e é necessário aguardar as instruções recebidas do *gsmSCF* para os dois serviços. Desta forma, este cenário é o que implica maiores tempos de atraso e maior consumo de recursos devido à complexidade inerente de activar dois serviços assíncronos que influenciam a sessão em causa.

Em chamadas do tipo MT o IM-SSF apenas activa o serviço VT_IM_CSI.

4.3 *Testes de desempenho*

Depois de terem sido efectuados os testes funcionais com sucesso, os testes de desempenho permitem validar a estabilidade e a eficiência do sistema.

As medidas que se pretendem avaliar com os testes de desempenho são:

- *Tempo de resposta* – tempo que demora o sistema desde que recebe a mensagem SIP "INVITE" até a enviar para o destino correcto.
- *Sessões perdidas* – percentagem de sessões perdidas.
- *Consumo de memória* – memória base consumida por módulo e o incremento de memória consumido pelo aumento do número máximo de sessões que o sistema pode processar em simultâneo.
- *Consumo de CPU* – percentagem média de processador que o sistema consome para vários ritmos de chegadas de sessões.

No início do desenvolvimento do sistema, foi especificado como objectivo que o sistema deveria suportar até 20000 sessões simultâneas e até 100000 clientes registados. Para puder assegurar estes valores, efectuaram-se inicialmente vários testes para ajustar os parâmetros configuráveis de cada módulo. Note-se que para atingir um funcionamento óptimo, estes parâmetros não podem ser nem demasiado pequenos nem demasiado grandes. Se forem demasiado pequenos, devido aos tempos de inactividade das threads do sistema, este passa demasiado tempo inactivo e não consegue processar em tempo útil (alguns segundos) as várias mensagens da sessão, ou então atrasa muito o processamento das várias mensagens. Se os valores dos parâmetros configuráveis forem demasiado grandes, o sistema tende a entrar pouco tempo em inactividade aumentando o consumo de CPU. No limite as várias threads entram em concorrência pelo processador atrasando o processamento das mensagens das sessões. Note-se também que há certas relações entre parâmetros que devem ser respeitadas. Por exemplo, a relação entre as mensagens SIP recebidas (NUMBER_MSG_GET_SIPQUEUIN) e enviadas (NUMBER_MSG_GET_SIPQUEUOUT) deve ser idêntica visto que para todos os

pedidos SIP temos sempre uma resposta. Após os vários testes preliminares executados, os parâmetros adoptados para o CNIMSSF são os que se mostram na Tabela 4-4.

Parâmetros configuráveis no CNIMSSF	*Configuração*
NSESSIONPROC	**120**
NCALLPROCIMCN	**60**
MAXNUMBEROFPROCESSCAP	**45**
MAX_SIP_POOLS	**15**
MAX_SIP_SENT	**15**
NUMBER_MSG_GET_INTQUEUIN	**60**
NUMBER_MSG_GET_INTQUEUOUT	**15**
NUMBER_MSG_GET_SIPQUEUIN	**15**
NUMBER_MSG_GET_SIPQUEUOUT	**15**
NUMBER_MSG_GET_CAPQUEUIN	**45**
usleep (μs)	**20000**
MaxNumberCalls	**20000**
Max_Clientes_Registered	**100000**

Tabela 4-4: Configuração para o módulo CNIMSSF

O módulo SIPSTACK possui um sistema próprio de gestão de memória. É neste contexto que surge a noção de *buckets*. Um *bucket* é uma espécie de contentor de memória onde cada pedaço de memória dentro de cada contentor tem um determinado tamanho. Assim, definindo vários contentores com diferentes tamanhos, pode-se criar um espaço de memória gerido pelo nosso sistema da mesma forma que o sistema operativo gere a sua memória. Por exemplo, se temos dois contentores, um que usa blocos do tamanho 128 *bytes* e o outro de 64 *bytes,* quando é necessário reservar 276 *bytes,* é preferível usar dois blocos de 128 *Bytes* e um de 64 do que 5 de 64 *Bytes* pois diminui-se o número total de blocos usado. Esta solução é implementada pela Trillium (empresa que forneceu a *stack* SIP desenvolvida em C), muito provavelmente porque o objectivo foi o de produzir um sistema independente da arquitectura e sistema operativo onde funciona. Desta forma, conseguem uniformizar a gestão de memória do seu software tornando-o mais independente da máquina onde é instalado. No caso da SIPSTACK, é usado um sistema de quatro (4) *buckets* onde o tamanho em *bytes* de um bloco de cada *bucket* é o apresentado na Tabela 4-5, entre parênteses, junto ao nome do

bucket. A SIPSTACK possui uma camada desenvolvida na PT Inovação que permite a comunicação desta com o CNIMSSF. Nesta camada existe uma thread que utiliza os parâmetros MAX_SIP_POOLS e MAX_SIP_SENDED para saber o número de mensagens SIP que são processadas no sentido de recepção (MAX_SIP_POOLS) e no sentido de envio (MAX_SIP_SENDED) em cada ciclo. Ao fim de um ciclo esta thread fica inactiva por um período indicado pelo parâmetro usleep (20ms). Os valores apresentados na Tabela 4-5 foram obtidos após a análise aos resultados dos testes de memória efectuados, testes estes que são apresentados na secção 4.3.2 .

Parâmetros configuráveis na SIPSTACK	*Configuração*
MAX_SIP_POOLS	35
MAX_SIP_SENDED	35
usleep(micro segundos)	20000
MaxNumberCalls	20000
Bucket 0 (80)	2400000
Bucket 1 (256)	500000
Bucket 2 (512)	165000
Bucket 3 (2048)	40000

Tabela 4-5: Configuração para o módulo SIPSTACK

Na Tabela 4-6 são apresentados os valores de configuração encontrados para o módulo CSGW seguindo uma filosofia semelhante à utilizada para o CNIMSSF. No CSGW a única relação é entre os pedidos e respostas. Na pior situação, um pedido CAP (IDP) pode originar três (3) respostas consecutivas, assim, o parâmetro MaxNumberOfProcessResponses, que indica quantas respostas CAP se devem processar num ciclo, é três vezes superior ao número de pedidos CAP que devem ser processados por ciclo (parâmetro MaxNumberOfProcessRequests). O CSGW à semelhança dos outros módulos possui um parâmetro de configuração que permite configurar o tempo de inactividade das suas threads (SleepMain).

Parâmetros configuráveis no CSGW	*Configuração*
MaxNumberOfProcessRequests	20
MaxNumberOfProcessResponses	60
SleepMain (ms)	20

Tabela 4-6: Configuração para o módulo CSGW

4.3.1 *Tempo de* resposta e Sessões perdidas

Em termos de sessões SIP, as mais frequentes são sem dúvida as sessões de chamada, sessões de cancelamento (toques) e sessões onde o destino se encontra ocupado.

Na Figura 4-3 é apresentado um fluxo onde são distinguidas duas sessões SIP e uma sessão CAP. A sessão SIP proveniente do chamador (10.112.48.115:5060) encontra-se a verde e a sessão do chamado (10.112.48.115:5061) a azul. O preto distingue a sessão CAP.

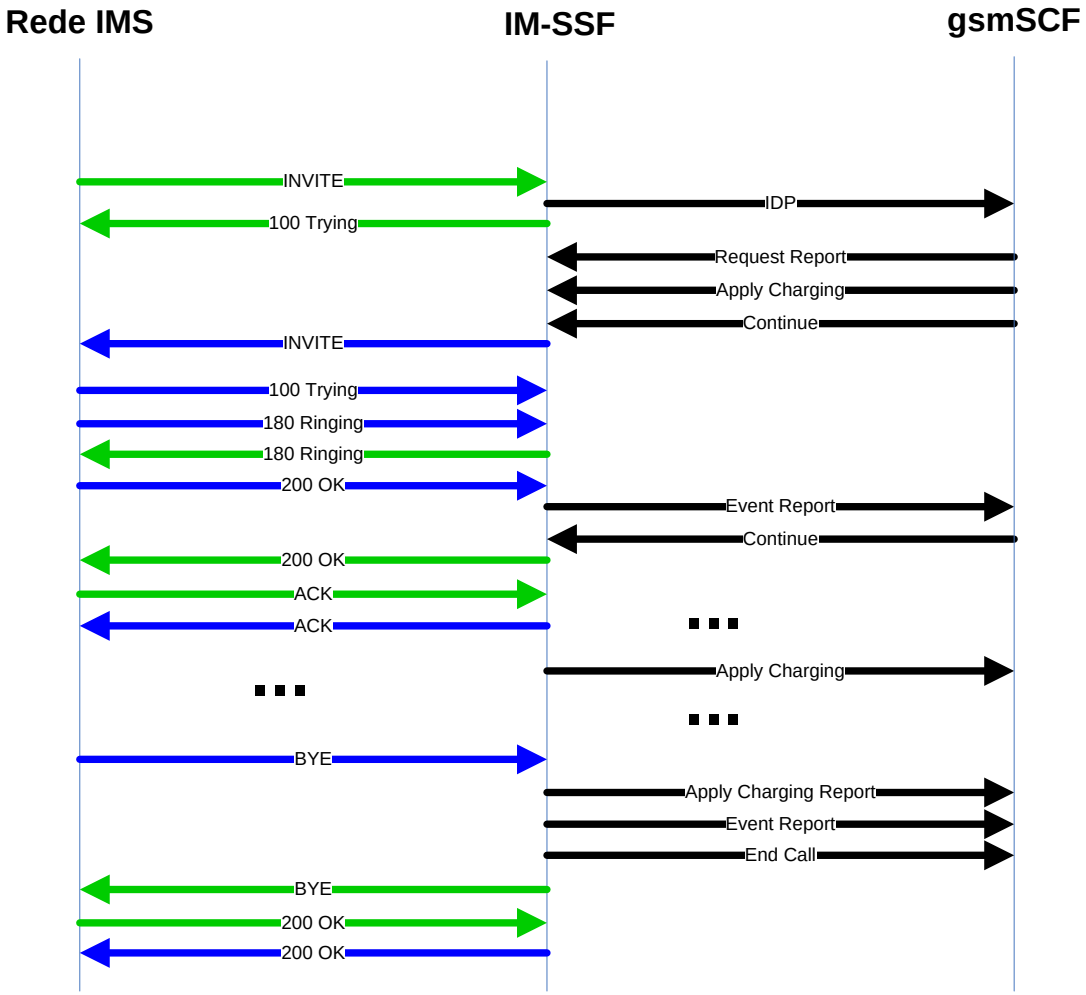

Figura 4-3: Fluxo de estabelecimento de uma sessão

No fluxo apresentado na Figura 4-3 encontram-se tanto as mensagens SIP como as mensagens CAP originadas e recebidas pelo IM-SSF.

Os eventos armados (os eventos são armados pela mensagem *"Request Report BCSM Event"*) e accionados neste fluxo são do tipo de notificação. Isto significa que o IM-SSF quando detecta que um evento deve ser reportado envia-o usando a mensagem *Event Report* e não necessita de aguardar nenhuma resposta proveniente do *gsmSCF*.

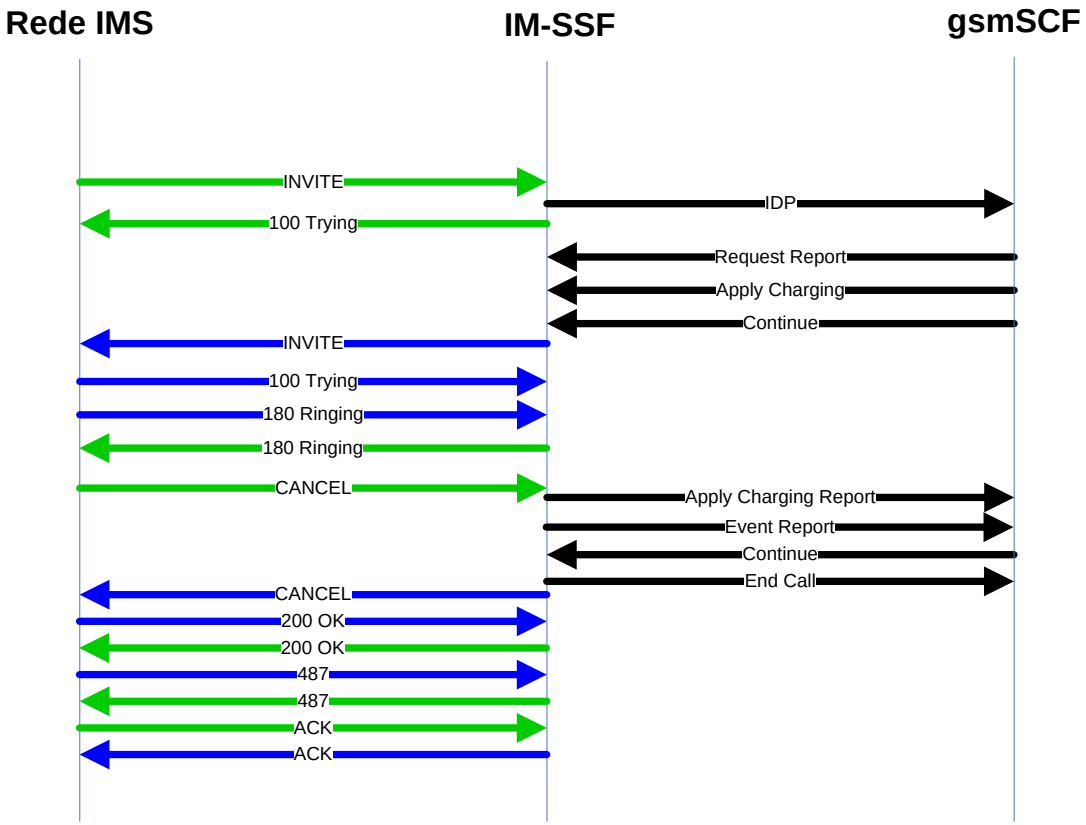

Figura 4-4: Fluxo de sessões canceladas

Na Figura 4-4 é apresentado o fluxo de mensagens usado nas sessões canceladas. Este tipo de sessões caracteriza-se pelo não estabelecimento da sessão. O chamador (10.112.48.115:5062) após efectuar o pedido envia uma mensagem, "Cancel", que como o nome indica cancela o pedido. O cancelamento é passado ao chamado (10.112.48.115:5063). É de notar que como o IM-SSF é um B2BUA não necessita de aguardar a confirmação do chamado para aceitar o término do pedido.

Nas mensagens CAP é de notar que ao *Event Report* final o *gsmSCF* responde com uma mensagem *Continue*. Isto deve-se ao facto do evento de cancelamento estar armado como interrupção, por outras palavras após o envio deste evento para o *gsmSCF* o IM-SSF é obrigado a aguardar uma resposta para poder prosseguir com a sessão.

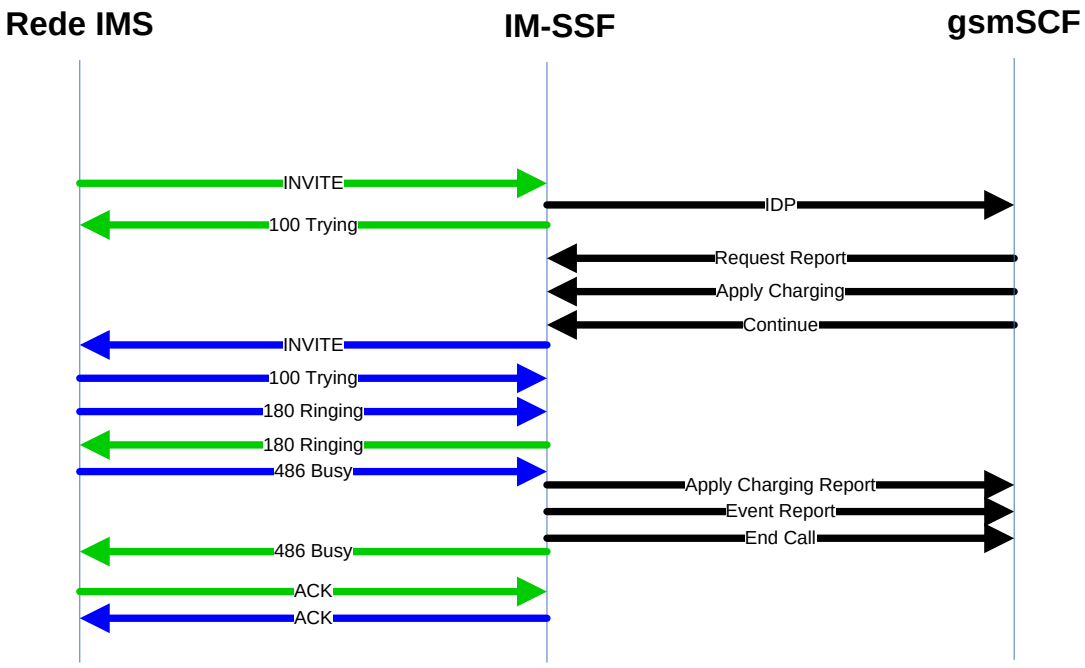

Figura 4-5: Fluxo de sessões rejeitadas pelo destino

O fluxo de sessões rejeitadas pelo destino (10.112.48.115:5065) é semelhante ao do cancelamento com a diferença de o término ter sido iniciado pelo destino. Um possível serviço que pode ser fornecido nestes casos seria o serviço de após a falha de tentativa de sessão esta ser encaminhada para outro destino. Isto pode ser feito armando o evento como interrupção e o *gsmSCF* ao receber este evento responderia com um *Connect* permitindo ao IM-SSF efectuar um *reconnect* para outro destino fornecido no *Connect*. Desta forma é possível implementar o correio de voz que hoje em dia já existe em GSM.

Com estas três situações desenharam-se três cenários capazes de simular as mensagens correspondentes de cada cenário. Recorrendo a um gerador SIP *freeware*, baseado em XML ([SIPp]) que permite configurar os fluxos SIP apresentados, conseguiu-se enviar as mensagens relativas a cada um dos três cenários, em simultâneo, com um ritmo de 20 sessões por segundo para cada cenário, o que no IM-SSF perfaz um ritmo aproximado de 60 sessões por segundo. É de salientar que este ritmo é gerado em rajada sendo este o pior caso para o sistema, visto que num caso real o tráfego é mais distribuído ao longo do tempo. Cada um dos cenários efectuou cerca de 100000 sessões em cada corrida. Efectuaram-se 10 corridas nestas condições com a duração total de aproximadamente 14

horas, o que permitiu obter os resultados apresentados na Tabela 4-7 e Tabela 4-8. O cenário de sessões estabelecidas foi configurado para estas terem uma duração de cerca de 120 segundos.

Parâmetro	*Descrição*	*Número de sessões*	*Valor médio (ms)*
T1	Tempo desde a recepção de um *"INVITE"* até ao seu envio pelo IM-SSF	300000	**456**
T2	Tempo desde o envio de um "INVITE" até à chegada do "200 OK"	100000	**963**
T3	IDP até à primeira mensagem CAP recebida no CNIMSSF	300000	**127**

Tabela 4-7: Tempos de resposta

O tempo T2 apresentado na Tabela 4-7 foi contabilizado apenas para o cenário de estabelecimento de sessão.

Em relação aos tempos de resposta tendo em conta que o tempo máximo de estabelecimento de sessão aceitável pelas operadoras é na ordem dos segundos, conclui-se que o tempo de 456ms (T1) é aceitável. No entanto é um valor que poderá ser melhorado. Embora não existam dados para o comprovar, foi verificado empiricamente que uma grande parte do tempo T1 é perdido no módulo SIPSTACK.

No mesmo teste onde se mediram os tempos de resposta foram também recolhidos dados sobre as sessões, que são apresentados na Tabela 4-8.

		Sessões mistas	*Chamada*	*Cancel*	*Busy*
	Corrida	*Total*	*Sessões Processadas*	*Sessões Processadas*	*Sessões Processadas*
Sessões com sucesso	1	300000	100000	100000	100000
Sessões com sucesso	2	300000	100000	100000	100000
Sessões com sucesso	3	300000	100000	100000	100000
Sessões com sucesso	4	299979	99999	100000	99980
Sessões com sucesso	5	300000	100000	100000	100000
Sessões com sucesso	6	300000	100000	100000	100000
Sessões com sucesso	7	300000	100000	100000	100000
Sessões com sucesso	8	300000	100000	100000	100000
Sessões com sucesso	9	300000	100000	100000	100000
Sessões com sucesso	10	300000	100000	100000	100000
Total Sessões/corrida		300000	100000	100000	100000
Número de corridas		10	10	10	10
%perdas		0,0007	0,0001	0	0,002

Tabela 4-8: Valores obtidos para sucesso de chamadas

Em relação às sessões perdidas, as perdas encontradas são cerca de 0,0007% o que significa que o nível de confiança em que o módulo consiga processar a sessão com sucesso é cerca de 99,9993% o que significa que em 1 milhão de sessões apenas 7 são perdidas para um ritmo de sessões por segundo de 60, o que é um resultado razoavelmente bom. Como as máquinas onde foram efectuados estes testes não são única e exclusivamente para o uso deste sistema suspeita-se que o insucesso das sessões na quarta corrida se deva ao uso da máquina por outros sistemas que podem num determinado momento, ter provocado a escassez de recursos no sistema, obrigando-o a perder algumas sessões.

4.3.2 Consumo de memória

Para se encontrarem os valores de crescimento de memória em função do número de sessões activas foi executado um fluxo de estabelecimento de sessão (Figura 4-3) onde sabe-se à partida quais as mensagens transaccionadas e o seu tamanho (isto permite o

cálculo de largura de banda utilizada). O fluxo foi executado 7 vezes fazendo variar o tempo de duração da sessão (tempo entre o "ACK" e o envio do "BYE").

O fluxo de estabelecimento de sessão é definido na Tabela 4-9 assim como o tamanho para cada mensagem.

	Ramo 1	*Ramo 2*
Fluxo	Tamanho (*bytes*)	Tamanho (*bytes*)
"INVITE"	1147	946
"100 Trying"	339	318
"180 Ringing"	632	319
"200 OK" ao "INVITE"	827	586
"ACK"	1025	451
"BYE"	884	397
"200 OK" ao "BYE"	487	320
Total	5341	3337

Tabela 4-9: Tamanho das mensagens usadas no fluxo de estabelecimento de sessão

Na Tabela 4-9 são distinguidos dois ramos pois o IM-SSF como B2BUA define dois ramos (duas sessões SIP relacionadas pelo IM-SSF), um do chamador para o IM-SSF (Ramo 1) e outro do IM-SSF para o chamado (Ramo 2). A diferença de tamanho entre os dois ramos deve-se ao facto do IM-SSF remover toda a informação SIP considerada supérflua quando inicia o Ramo 2.

Executou-se o fluxo da Tabela 4-9 e efectuaram-se várias corridas para o mesmo ritmo e para durações de sessão diferentes. Desta forma conseguiram-se os valores para os buckets em função das sessões activas. Os dados obtidos são apresentados na Tabela 4-10.

Valores Configurados	Sessões por segundo	1	50	50	50	50	50
	Duração (Seg.)	1	1	20	40	60	80
Valores Medidos	Número de sessões Activas (valor médio)	1	75	1049	2048	3046	4044
	Bucket 0 (80)	110	130225	250884	368422	502306	628934
	Bucket 1 (256)	11	34608	57821	79663	105659	129771
	Bucket 2 (512)	21	11459	19191	26559	35204	43198
	Bucket 3 (2048)	6	2001	3922	5794	7925	9923
Valores Calculados	Atraso médio por sessão (ms)	0	500	980	960	920	880

Tabela 4-10: Valores de utilização de memória da SIPSTACK

O atraso médio por sessão é calculado fazendo a divisão do número de sessões activas pelas sessões por segundo, subtraindo a duração da sessão ($\frac{\text{Número de sessões activas}}{\text{(sessões por segundo)}} - \text{Duração}$). Note-se que este atraso é o atraso sofrido no sistema pela sessão, desde que a sessão é criada (primeiro "INVITE") até ao seu término ("200 OK" ao "BYE").

Através dos valores obtidos na Tabela 4-10 conseguiu-se obter o gráfico apresentado na Figura 4-6 que representa o crescimento dos buckets (o crescimento do bucket é directamente proporcional ao crescimento de memória utilizada pelo processo) em função do número de sessões activas.

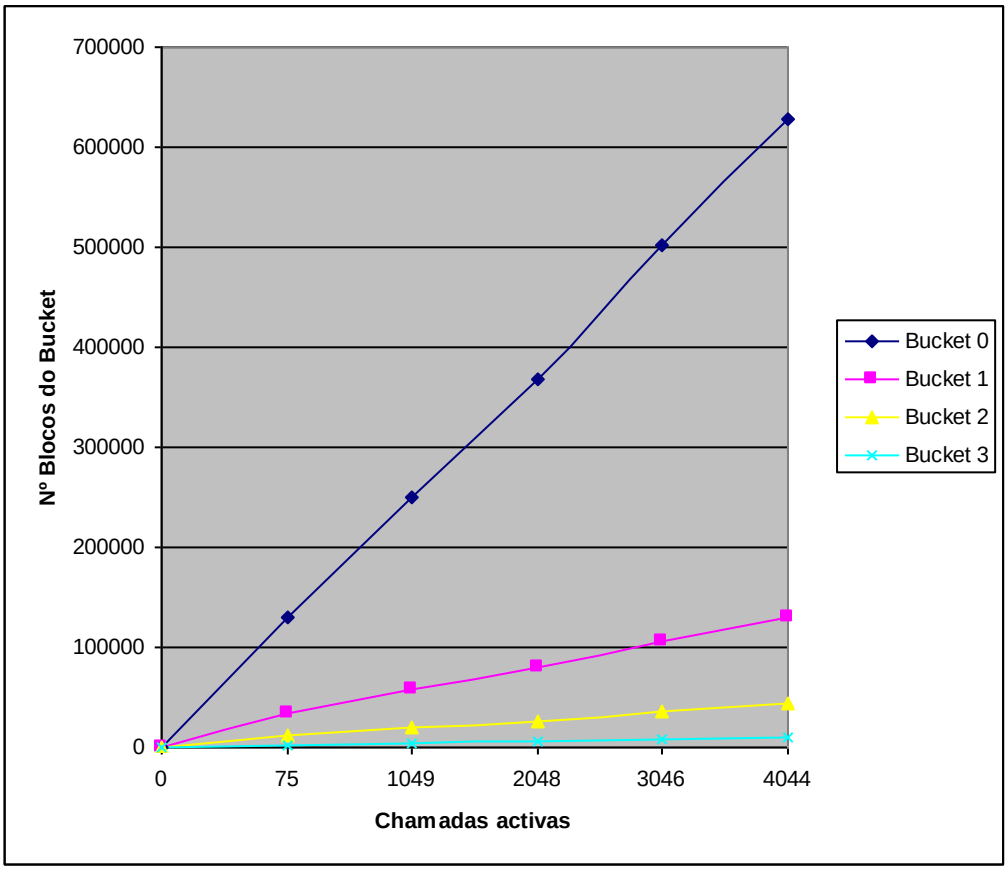

Figura 4-6: Utilização de memória da SIPSTACK

Usando a regressão linear do *Microsoft Excel 2003* que tenta aproximar uma recta sobre os valores obtidos conseguiram-se rectas lineares que são muito próximas dos valores obtidos. Através destas rectas consegue-se uma estimativa que permite configurar os valores para os *buckets* de acordo com a sua utilização. Desta forma para se ter cerca de 20000 sessões activas em simultâneo na SIPSTACK devem-se configurar os *buckets* com os valores apresentados na Tabela 4-11. O total de memória é calculado somando o tamanho de cada bucket em bytes. Para calcular o tamanho de cada bucket é só multiplicar o número de elementos de bucket pelo tamanho de cada elemento em bytes $((80 * 2377350) + (256 * 484563) + (512 * 161434{,}5) + (2048 * 37565{,}6))$.

bucket 0 (80)	2377350
bucket 1 (256)	484563
bucket 2 (512)	161434,5
bucket 3 (2048)	37565,6
Total MEM(MB)	473,825

Tabela 4-11: Configuração dos buckets para 20000 sessões activas

Na Tabela 4-12 são apresentados os valores de utilização de memória para cada módulo utilizado. Nesta tabela são apresentados os valores de memória na forma *Virt* e *Res*. O *Virt* é toda a memória acessível pelo módulo, isto inclui memória de bibliotecas partilhadas, memória alocada pelo módulo, memória mapeada pelo módulo em ficheiros do sistema. O *Res* é a memória residente que é uma representação fiel da memória física que o módulo ocupa.

Nº Sessões	***1***		***100***		***1000***	
Memória (MB)	Virt	Res	Virt	Res	Virt	Res
CNIMSSF	58,20	12,00	9,11	13,00	68,33	21,00
CSGW	60,47	2,68	61,02	3,18	65,62	7,83

Tabela 4-12: Utilização de memória pelos módulos CNIMSSF e o CSGW

Através da análise da Tabela 4-12 consegue-se extrapolar os dados referentes à memória base necessitada por cada módulo e o seu acréscimo (Δ) por cada sessão activa. Com isto consegue-se construir a Tabela 4-13.

	Base		***Δ***	
Memória (MB)	***Virt (MB)***	***Res (MB)***	***Virt(MB)***	***Res (MB)***
SIPSTACK	541,00	354,00	0,00	0,00
CNIMSSF	58,19	11,99	0,01	0,01
CSGW	60,47	2,67	0,01	0,01

Tabela 4-13: Valores de memória base e acréscimo por sessão activa

Observando os valores de memória obtidos e seu acréscimo, conclui-se que o sistema para 20000 sessões simultâneas necessita de perto de 1 giga de memória. Sendo que metade é ocupado pela SIPSTACK, que faz a descodificação do SIP. Embora possa parecer um valor elevado, para as máquinas utilizadas nos testes (2 GB) para um operador é um valor razoável. Este sistema foi instalado num cliente, em duas máquinas com 8 (GB) cada.

4.3.3 Consumo de CPU

Para medir os valores de CPU foram efectuadas novas corridas a vários ritmos de sessões por segundo. Usando esta metodologia obtiveram-se os resultados apresentados na Tabela 4-14.

Sessões/Seg.	*50*	*100*	*150*	*200*
SIPSTACK (%)	12,7	21,6	36,1	53,7
CNIMSSF (%)	4,9	9,8	17,6	26,3
CSGW (%)	2,9	5,9	10,7	17,5

Tabela 4-14: Utilização de CPU em função do ritmo de sessões por segundo

Os dados obtidos para o consumo de CPU são bons pois os requisitos do cliente onde o sistema foi instalado são de ritmos inferiores às 50 sessões por segundo. Como se verifica, até 50% de consumo de CPU, que é uma carga aceitável numa máquina de produção, o sistema poderá processar sessões a ritmos de 150 sessões por segundo.

Conclui-se que o sistema é bastante eficiente no que diz respeito ao consumo de CPU.

Capítulo 5 - Conclusões

O primeiro objectivo desta dissertação foi o estudo das normas vigentes que especificam o IM-SSF. Este estudo prendeu-se fundamentalmente com a análise da norma [3GPP TS 23.278 V6.1.0]. Contudo, por muito boa que a norma seja, foi também necessário perceber como o IM-SSF se enquadra na arquitectura IMS e como interage com os seus outros componentes de forma a fornecer os serviços a que se propõe. É neste contexto que surge o segundo objectivo que foi o de estudar os protocolos que suportam as interfaces entre o IM-SSF e as entidades a si associadas. Os protocolos analisados foram o SIP, CAP, MAP e Diameter. De entre os protocolos referidos, os mais importantes foram o SIP e o CAP pois são estes que permitem o estabelecimento de sessões. Os outros dois protocolos (MAP e Diameter) são apenas utilizados na comunicação com o HSS para a obtenção dos dados relativos aos utilizadores. A implementação do IM-SSF desenvolvida neste trabalho baseia-se na utilização de ficheiros de configuração para leitura dos dados de utilizador mais relevantes, que doutra forma poderiam ser obtidos do HSS. Mesmo com as limitações impostas pelo facto de se usarem ficheiros de configuração (em vez da comunicação com o HSS), a implementação desenvolvida é uma mais-valia para redes onde não exista HSS. Com isto chega-se ao terceiro objectivo, o de implementar o módulo IM-SSF. Como a implementação segue a norma [3GPP TS 23.278 V6.1.0], a alteração e mesmo leitura do código torna-se mais simples. Mesmo o

facto de a norma ser bastante boa em detalhes e fluxo de sessões, ainda não está fechada (à data da escrita desta dissertação). Não obstante algumas melhorias que podem ser efectuadas à solução desenvolvida, os testes de desempenho provaram ser uma solução robusta e eficaz. Assim, avaliando os objectivos propostos para esta dissertação, pode-se concluir que estes foram globalmente cumpridos.

5.1 *Software e soluções adoptadas*

Desde o desenvolvimento do software, às soluções criadas para resolver determinadas situações, até aos testes efectuados, provou-se o bom funcionamento do sistema face às especificações normativas. Com os testes de desempenho, provou-se que a solução é suficientemente robusta e versátil para fazer face às necessidades efectivas de um ambiente de negócio.

De entre as várias soluções adoptadas, a mais importante e relevante em termos de versatilidade de negócio foi a solução de ler os dados necessários para o prosseguimento de uma sessão a partir de ficheiros. Esta solução abre o leque de opções de mercado, pois sendo o IMS uma rede relativamente recente, é muito frequente encontrarem-se operadores que apenas possuem parte da rede e normalmente o HSS, quando existe, não tem interface MAP estritamente necessária para obter os valores imprescindíveis ao IM-SSF. A solução desenvolvida baseou-se em construir o equivalente aos CSI obtidos por MAP em três ficheiros com a limitação de não ser possível configurar um serviço por cliente fornecendo apenas o mesmo serviço a todos os clientes.

5.2 *Importância no IMS*

O IM-SSF é de extrema importância no IMS pois permite uma interligação da rede IMS com a lógica de serviço que existe actualmente. Com isto, a rede IMS consegue ser inserida pelos operadores sem qualquer impacto na lógica que efectivamente lhes permite ganhar dinheiro. É assim possível garantir que o seu negócio continua a funcionar normalmente não necessitando de grandes alterações à sua organização e posicionamento no mercado. Com a entrada efectiva do IMS como parte integrante do

núcleo do operador o IM-SSF ainda assim pode ser uma mais-valia pois permite que todos os mecanismos de controlo e desenho de lógicas de controlo continuem a ser válidos.

5.3 *Sugestões para o futuro*

Como em todos os sistemas semelhantes, é sempre possível serem efectuadas melhorias ao sistema, para serem corrigidos hipotéticos problemas e até para se acrescentar algumas funcionalidades para as quais o sistema não se encontra preparado.

Como o IM-SSF se trata de um software baseado em sessões, é sempre necessário existir alguns mecanismos para o término destas sessões. O software desenvolvido já possui alguns mecanismos mas estes são sempre baseados em informação recebida pelas suas interfaces. Embora nos testes efectuados não tenha surgido a situação de as sessões existentes ficarem presas e não serem libertadas não quer dizer que esta situação não possa acontecer. Ou seja, é sempre possível que um elemento que comunica com o IM-SSF não respeite a norma SIP levando a que uma sessão iniciada no IM-SSF não seja terminada. Para eliminar este tipo de possibilidade pode-se incorporar um temporizador que ao fim de algum tempo de inactividade SIP da sessão em causa, provoque o término dessa sessão. Com a regulação deste temporizador na ordem das horas, conseguem-se eliminar as eventuais situações de sessões presas. Esta medida terá pouco impacto nos clientes pois a maioria das sessões tem uma duração inferior a uma hora.

Embora se notem grandes deficiências na norma quando o IM-SSF tem de interagir com o MRFC é possível efectuar algumas alterações no IM-SSF que permitam um fluxo correcto de interacção com o MRFC no início de uma sessão, permitindo disponibilizar, por exemplo, o serviço de toque de anúncio no inicio da sessão.

Referências

[3GPP TS 23.002 V6.10.0] 3rd Generation Partnership Project, Technical Specification Group Services and Systems Aspects; (2005-12) "Network architecture"; (Release 6)

[3GPP TS 23.218 V6.4.0] 3rd Generation Partnership Project, Technical Specification Group Core Network IP Multimedia (IM) session handling; (2006-06) "IM call model"; Stage 2 (Release 6)

[3GPP TS 23.228 V6.8.0] 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; (2004-12) "IP Multimedia Subsystem (IMS)";
Stage 2 (Release 6)

[3GPP TS 23.278 V6.1.0] 3rd Generation Partnership Project, Technical Specification Group Core Network and Terminals (2005-06) "Customised Applications for Mobile network Enhanced Logic (CAMEL) Phase 4"; Stage 2; IM CN *Interworking* (Release 6)

[3GPP TS 24.229 V6.14.0] 3rd Generation Partnership Project; Technical Specification Group Core Network and Terminals; (2007-03) "IP multimedia call control protocol based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP)";
Stage 3 (Release 6)

[3GPP TS 29.278 V6.1.0] 3rd Generation Partnership Project, Technical Specification Group Core Network (2005-03) "Customised Applications for Mobile network Enhanced Logic (CAMEL) Phase 4; CAMEL Application Part

(CAP) specification for IP Multimedia Subsystems (IMS)" (Release 6)

[AAA Jaques] Aboba, B., Calhoun, P., Glass, S., Hiller, T., McCann, P., Shiino, H., Zorn, G., Dommety, G., Perkins, C., Patil, B., Mitton, D., Manning, S., Beadles, M., Walsh, P., Chen, X., Sivalingham, S., Hameed, A., Munson, M., Jacobs, S., Lim, B., Hirschman, B., Hsu, R., Xu, Y., Campbell, E., Baba, S. and E. Jaques, "Criteria for Evaluating AAA Protocols for Network Access", RFC 2989, November 2000.

[ALP Samuel] Mark Mitchell, Jeffrey Oldham, Alex Samuel "Advance Linux Programming" , USA, New Riders Publishing, 1ª ed., 2001, 340p

[Diameter Perkins] P. Calhoun, C. Perkins, "Diameter Mobile IP Application", Work in Progress.

[IMS Mikka] KHARTABIL (Hisham), MAYER (Georg), NIEMI (Aki) e POIKSEKA (Mikka), "The IMS IP Multimedia Concepts and Services in the Mobile Domain", England, Wiley, 1ª ed., 2004, 419p

[NASREQ Haag] P. Calhoun, W. Bulley, A. Rubens, J. Haag, "Diameter NASREQ Application", Work in Progress

[OO Meyer] Bertrand Meyer, "Object-Oriented Software Construction", USA, Prentice Hall, 1997, 1296p

[RFC 3261] (2002) SIP: Session Initiation Protocol, Junho 2002. URL: http://www.ietf.org/rfc/rfc3261.txt [1 Junho 2007]

[RFC 3588] (2003) Diameter Base Protocol, Setembro 2003, URL: http://www.ietf.org/rfc/rfc3588.txt [1 Junho 2007]

[SIP RFCs] (2006) SIP RFCs and Drafts,

URL: http://www1.cs.columbia.edu/sip/drafts.html [17 Outubro 2006]

[SIPp] (2007) SIPp URL: http://sipp.sourceforge.net/ [1 Junho 2007]

[SS7 Teixeira de Sousa] Henrique Teixeira de Sousa, "SINALIZAÇÂO nº 7 teoria e prática", Portugal, João Azevedo Editor, 1ª ed, 2005, 351p

[TECH-INVITE] (2007) Tech Invite, SIP Portal URL: http://www.tech-invite.com/ [23 Abril 2007]

[TRILLIUM] (2007) Continuous Computing URL:http://www.ccpu.com/ [16 Fevereiro 2007]